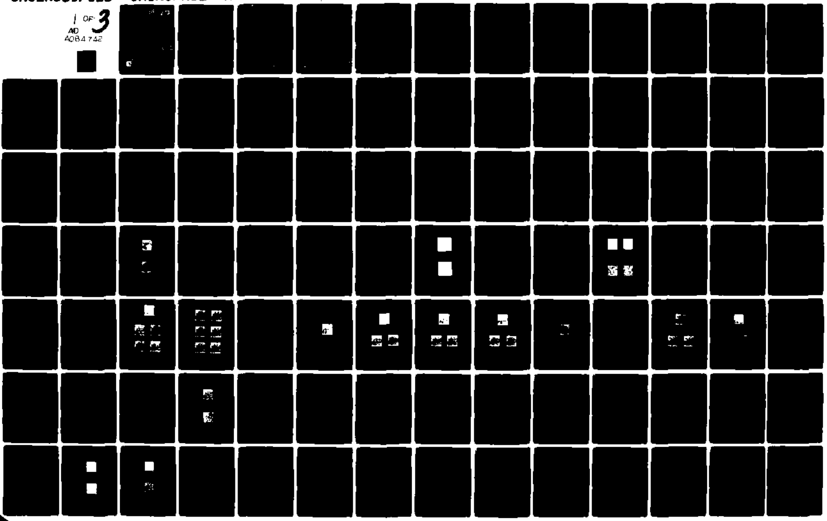
AD-A084 742   KANSAS UNIV/CENTER FOR RESEARCH INC LAWRENCE REMOTE --ETC F/G 9/2
INVESTIGATION OF EXTREMA IN DIGITAL IMAGES FOR TEXTURE ANALYSIS--ETC(U)
MAR 79 A SINGH, R M HARALICK DAAK70-77-C-0156
UNCLASSIFIED CRINC/RSL-TR-357-1 ETL-0210 NL

1 OF 3
AD
A084 742



CRINC REMOTE SENSING LABORATORY

ETL-0210

INVESTIGATION OF EXTREMA IN DIGITAL IMAGES FOR TEXTURE ANALYSIS

Final Report

Amrendra Singh
Robert M. Haralick

Remote Sensing Laboratory
Center for Research, Inc.
University of Kansas
Lawrence, Kansas 66045

RSL TR 357-1

March 1979

Prepared for:

U.S. Army Engineer Topographic Laboratories
Fort Belvoir, Va. 22060

THE UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.
2291 Irving Hill Drive—Campus West Lawrence, Kansas 66045

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered) (14) CRINC / RSL-TR-357-1

(17) REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER (1) ETL-0210	2. GOVT ACCESSION NO. AD-A084 142	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) INVESTIGATION OF EXTREMA IN DIGITAL IMAGES FOR TEXTURE ANALYSIS		5. TYPE OF REPORT & PERIOD COVERED Contract Report Sept 77 - Mar 79
		6. PERFORMING ORG. REPORT NUMBER RSL TR 357-1
7. AUTHOR(s) A. Singh R. Haralick (10) [illegible] / Singh Robert M. / Haralick		8. CONTRACT OR GRANT NUMBER(s) DAAK70-77-C-0156 new
9. PERFORMING ORGANIZATION NAME AND ADDRESS Center for Research, Inc. University of Kansas Lawrence, Kansas 66045		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS (12) 196
11. CONTROLLING OFFICE NAME AND ADDRESS US Army Engineer Topographic Laboratories Ft. Belvoir, VA 22060		12. ~~REPORT DATE~~ (11) March 1979
		13. NUMBER OF PAGES 192
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) (9) [illegible] Sept. Sep 77-Mar 79		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Texture	Digital Images	Image Extrema
Extrema	Image Segmentation	Reachability Sets
Pattern Recognition	Clustering	Extrema Density

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report investigates the use of extrema (relative minima and maxima) graytones of a digital image, as primitives for texture analysis. Texture in images may be described in terms of extrema density as well as by the attributes of the extrema primitive. The structure of the reachability set of the extrema is examined. Included are algorithms to extract these as well as their properties. The report also demonstrates how image segmentation can be achieved on textured regions by the use of primitives such as the reachability sets.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

410 11

THE UNIVERSITY OF KANSAS CENTER FOR RESEARCH, INC.
2291 Irving Hill Drive—Campus West
Lawrence, Kansas 66045

Telephone: 913-864-4836

ETL-0210

INVESTIGATION OF EXTREMA IN DIGITAL IMAGES FOR TEXTURE ANALYSIS

Final Report

Amrendra Singh
Robert M. Haralick

Remote Sensing Laboratory
Center for Research, Inc.
University of Kansas
Lawrence, Kansas 66045

RSL TR 357-1

March 1979

Prepared for:

U.S. Army Engineer Topographic Laboratories
Fort Belvoir, Va. 22060

PREFACE

This report was prepared by the University of Kansas Center for Research, Inc. for the U.S. Army Engineering Topographic Laboratories (ETL) under contract DAAK70-77-C-0156.

The Contracting Officer's Representative during this period was Mr. L. P. Murphy.

The authors would like to express their appreciation to Chris Hlavka, Ryan Stansifer and Kirk Kitchen for all their help in programing and performing experiments for this project.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (cont.)

LIST OF ILLUSTRATIONS (cont.)

LIST OF ILLUSTRATIONS (cont.)

1.0 INTRODUCTION

In the past few years texture has proved to be a powerful tool in the fields of pattern recognition, remote sensing, and digital scene analysis. The earlier works on image segmentation were based primarily on tonal or gray level properties, mainly because they were easier to understand and implement. The use of texture analysis as an aid to segmentation came about because of the limited success with the earlier techniques. Also, researchers soon realized that not only does texture play an important role in human visual perception, but it is also intrinsically linked to the basic structure of a scene. This last point is important, and will be explored in more detail later.

Despite the extensive amount of research in the field, texture is still poorly understood because a precise definition of texture is elusive. There are many descriptive terms available, e.g., coarse, fine, granulated, linear, but nothing that captures texture properties precisely. In other words, our vocabulary for texture is only qualitative and not quantitative. However, this has not deterred researchers in making use of texture features that they have found suitable for their work. A general semantic formulation which would connect different approaches is what is lacking.

Basically, texture is an organized area phenomenon (Haralick, 1979). When it is decomposable it has two basic dimensions. The first is for describing the primitives out of which the image texture is composed, and the second is for the spatial dependence or interaction between the primitives of an image texture. The first dimension is concerned with tonal primitives or local properties while the second deals in the spatial organization of the tonal primitives. Tonal primitives are regions with tonal properties. The tonal primitive can be evaluated in terms of its average tone, or maximum and minimum tone of its region. Its region is a maximally connected set of pixels having a given tonal property. Properties of regions can be size and shapes. The tonal primitive includes both its gray tone and tonal region properties.

An image texture is characterized by the number and types of its primitives and the spatial organization or layout of its primitives. The spatial organization may be random, may have a pairwise dependence of one primitive on a neighboring primitive, or may have a dependence of n primitives at a time. The dependence may be structural, probabilistic or functional (like a linear dependence).

These ideas of primitives and their attributes will be pursued further, as they support the hypothesis that digital image texture is characterized by the number and types of primitives and the spatial relationship between these primitives.

To objectively use the tone and textural pattern elements, the concepts of tonal and texture feature must be explicitly defined. With an explicit definition, we discover that tone and texture are not independent concepts. They bear an inextricable relationship to one another very much like the relation between a particle and a wave. There really is nothing that is only particle or only wave. Whatever exists has both particle and wave properties and depending on the situation, the particle or wave properties may predominate. Similarly, in the image context, tone and texture are always there, although at times one property can dominate the other and we tend to speak of only tone or only texture. Hence, when we make an explicit definition of tone and texture, we are not defining two concepts: we are defining one tone-texture concept.

The basic interrelationships in the tone-texture concept are the following. When a small-area patch of an image has little variation of tonal primitives, the dominant property of that area is tone. When a small-area patch has wide variation of tonal primitives, the dominant property of that area is texture. Crucial in this distinction are the size of the small-area patch, the relative sizes and types of tonal primitives, and the number and placement or arrangement of the distinguishable primitives. As the number of distinguishable tonal primitives decreases, the tonal properties will predominate. In fact, when the small-area patch is only the size of one resolution cell, so that there is only one discrete feature, the only property present is simply gray tone. As the number of distinguishable tonal primitives increases within the small-area patch, the texture property will dominate. When the spatial pattern in the tonal primitives is random and the gray tone variation between primitives is wide, a fine texture results. As the spatial pattern becomes more definite and the tonal regions involve more and more resolution cells, a coarser texture results (see Pickett, 1970).

According to Ehrich (1978), there are three problems of concern in texture analysis. These are listed in order of increasing difficulty. First, given a textured region, to which of a finite number of classes does it belong? Second, given a texture region, how can it be described? And third, given a scene, how can the boundaries between major textured regions be established?

The first problem is one of discrimination and is in the most part solvable. By suitably choosing the features relatively high success rates have been achieved. The second deals with the structure of texture which is usually very complex. The use of primitives and their attributes is one approach to this problem. Finally, the third deals with scene analysis and segmentation. It is the most difficult as it incorporates the first two problems and the interplay between them. In addition, parameters for levels of texture complexity and clustering have to be established. The human visual system succeeds with this third problem because of its excellent grouping mechanisms as well as the use of global and semantic information about the image. These latter techniques are difficult to implement on a computer, but future works in this field will have to incorporate them.

In this report we would like to look at some techniques that deal with the second and third problems mentioned above. These methods, even though they are powerful compared to previous techniques, are still limited. This is because none of these processes use semantic information in the image. Incorporation of this knowledge would be the next step in a more general and complex analysis scheme. One way to do this is given in Haralick and Shapiro (1979).

In order to give the methods investigated a proper perspective, we begin with a review of some basic approaches to texture analysis. This is done in Chapter 2. Chapter 3 discusses the techniques investigated. It contains a description of the primitives used and reasons for their selection. Also discussed are algorithms to compute these primitives. Chapter 4 discusses some experiments that were performed. In Chapter 5 we look at the attributes of these primitives and the general algorithm structure to extract these properties. This section also discusses the clustering philosophy and techniques used. Chapter 6 includes other related experiments carried out with the primitives of Chapter 3. This is followed by conclusions and suggestions for further work.

2.0 APPROACHES TO TEXTURE ANALYSIS

In the broadest sense there are two overlapping classes to texture analysis techniques: structural and statistical. Actually all techniques use both approaches to varying degrees. Structural usually defines the primitives to be used. These can be descriptions of patterns in a texture synthesis experiment or basic structural elements of an image. The statistical methods are brought into play to describe the relationships between the primitives. There are many ways to do this.

The classification of texture models given in Haralick (1979) consists of autocorrelation functions, optical transforms, digital transforms, autoregressive models, textural edgeness, gray level runlengths, syntactic, structural element, gray tone co-occurrence, and min/max extrema per unit area. The first three deal in some way with spatial frequency. Spatial frequency is related to texture because fine texture is rich in high spatial frequency, while coarse texture is rich in low spatial frequency. Thus, by working in the frequency domain texture information can be extracted. This same idea is used in measuring texture in terms of edgeness per unit area. Fine textures have many edges per unit area while coarse ones have fewer (Rosenfeld and Troy, 1970; Rosenfeld and Thurston, 1971). An experiment was performed which compared the edgeness per unit area to the approaches investigated. This is discussed later.

The structural element approach uses a matching procedure to detect the spatial regularity of shapes called structural elements in a binary image. In a process similar to template matching, the element is moved across the image. The image is eroded wherever a match does not occur. Textural features can be obtained by counting the number of cells eroded. Crucial to this approach are the shape of the element and direction of the scan. When the structural elements are single resolution cells, the information provided by this approach is the autocorrelation function. By using larger and more complex shapes, a more generalized autocorrelation can be computed. The main power of the structural element approach is its emphasis on shape, something that not many other approaches consider. So far, however, it has only been applied to binary images (Serra, 1973).

Autoregressive techniques involve using linear estimates of a pixel's gray tone given its neighborhood. Coarse textures then correspond to similar coefficients while fine texture areas will show a wide variance in the coefficients (Read and Jayaramurthy, 1972; McCormick and Jayaramurthy, 1975).

Gray level runlengths method by Galloway (1974) uses gray level runlengths as primitives and computes features in a manner similar to the spatial co-occurrence matrix method.

Syntactic approaches to texture deal mostly with synthesizing textures by phrase structure grammars (Lu and Fu, 1978).

The last two: gray tone co-occurrence and min/max extrema density are more related to the research done and thus are presented in more detail.

2.1 Gray Tone Co-Occurrence

Gray tone co-occurrence is the most explored technique in texture analysis today. First proposed by Rosenfeld and Troy (1970) and Haralick (1971), it has proved its worth by many other researchers. The basic reason for this is that it is a second-order statistic and thus takes into account second-order effects. As such, this kind of measure is called a strong texture measure.

To quantify our descriptions, some basic definitions are needed, and are given below.

Definition 2.1.1

Let $Z_r = \{1,2,\ldots,N_r\}$ and $Z_c = \{1,2,\ldots,N_c\}$ be a set of row and column indices. Let $G = \{1,2,\ldots,N_g\}$ be a set of gray levels. Then the digital image I, $I:Z_r \times Z_c \to G$ is a function which assigns to each pixel (picture element) a gray tone value. The pixel location is referred to by an ordered pair $(k,\ell) \in Z_r \times Z_c$ and its gray tone is $I(k,\ell)$.

A first-order statistic of the gray tones is any function of the number of times each different gray level occurs in the image. One such function is the gray level histogram of the image. Sometimes segmentation can be performed on the basis of a histogram. For example, if an image contained a set of pixel values from two different populations, the histograms would be bimodal. A threshold based on the valley in the histogram could yield the two different regions. Chow and Kaneko (1972) used this technique to find edges in lung X-ray images. However, segmentation based on histograms on very complex images is usually not very good.

The co-occurrence matrix, a second-order statistic, essentially measures the relative frequency P_{ij} of gray tone "i", occurring next to gray tone "j". The "next to" characterizes the spatial relationship between the primitives. It is usually specified by a distance between primitives and orientation.

Definition 2.1.2

Let G be the set of gray tones of I and R a binary relation pairing cells in $Z_r \times Z_c$ which are in the desired spatial relation, then the co-occurrence matrix is a function P, $P:G \times G \to (0,1)$ given by:

$$P(i,j) = \frac{\#\{((a,b),(c,d)) \in R \mid I(a,b) = i \text{ and } I(c,d) = j\}}{\#R}$$

That the co-occurrence matrix contains more information than the histogram is illustrated by the two simple images in Figure 2.1.1. While the histograms are the same, the co-occurrence matrices are not, and the images reflect the different texture structure.

Based on the co-occurrence matrices, Haralick et al. (1973) computed different features which were very successful in discriminating between textured images.

Another way to use the co-occurrence matrices is to generate a textural transform image based on them (Haralick, 1975). This is an image which indicates how common the texture pattern is in and around each resolution cell of the original image. It gives a "measure" of texture for each resolution cell. As lot of the work to be described involved generation of texture transforms of different types, so transforms based on co-occurrence counts were also created for comparison. Examples of these are given later. First let us examine the structure of this transform.

We wish to construct a transform image J such that the gray tone J(i,j) at resolution cell (i,j) in image J indicates how common the texture pattern is in and around resolution cell (i,j) of image I.

For analysis of the micro-texture, the gray tone J(i,j) can be a function of the gray tone I(i,j) and its nearest neighbors.

$$\begin{aligned} J(i,j) = f(&I(i-1,j-1),\ I(i-1,j),\ I(i-1,j+1),\ I(i,j-1),\ I(i,j),\\ &I(i,j+1),\ I(i+1,j-1),\ I(i+1,j),\ I(i+1,j+1)) \end{aligned}$$

Let us assume that this function f is an additive effect of horizontal, right diagonal, vertical, and left diagonal relationships. Then:

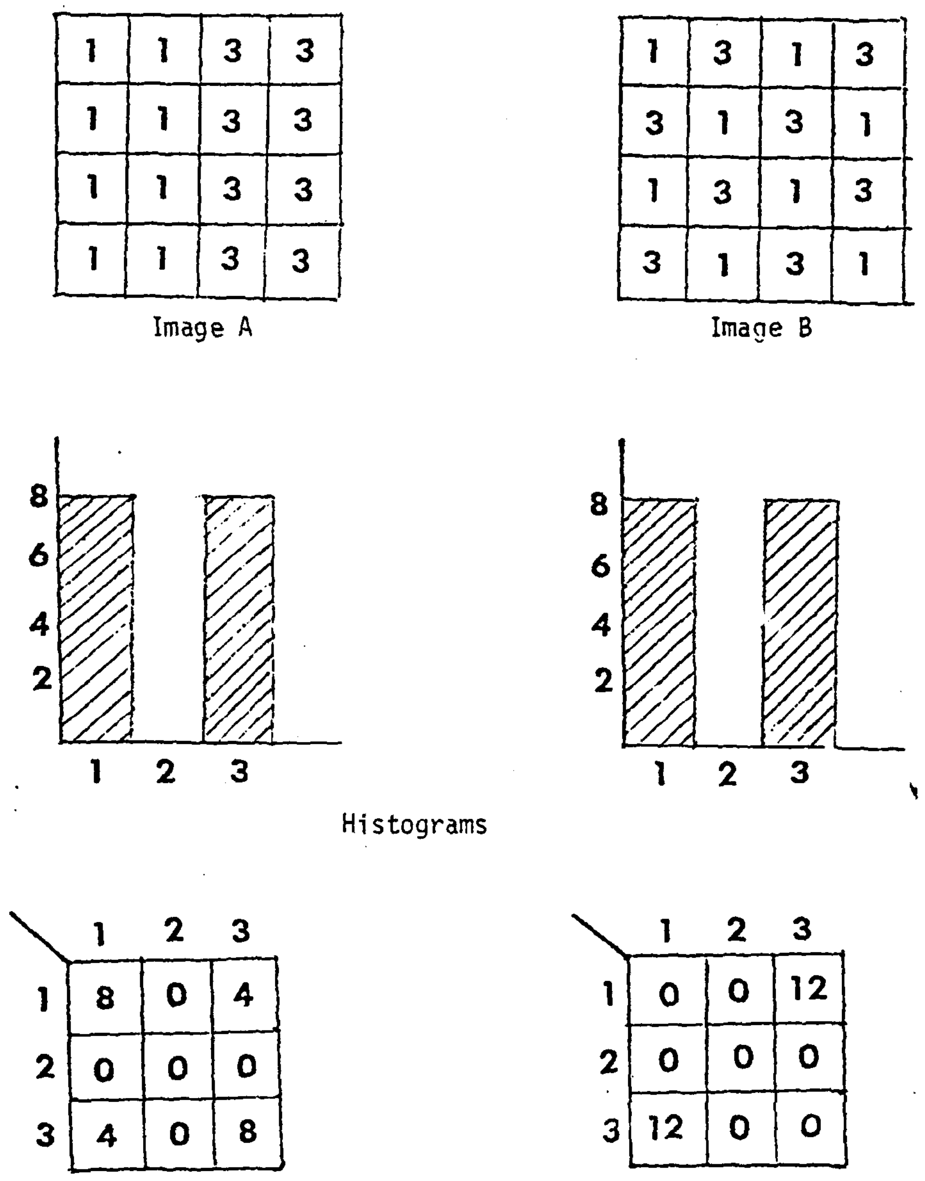


Figure 2.1.1. The difference between the first- and second-order statistics histograms and co-occurrence matrices is shown.

$$
\begin{aligned}
J(i,j) = {} & f_1(I(i,j-1),\ I(i,j),\ I(i,j+1)) && \text{(horizontal)}\\
& + f_2(I(i+1,j-1),\ I(i,j),\ I(i-1,j+1)) && \text{(right diagonal)}\\
& + f_3(I(i-1,j),\ I(i,j),\ I(i+1,j)) && \text{(vertical)}\\
& + f_4(I(i+1,j+1),\ I(i,j),\ I(i-1,j-1)) && \text{(left diagonal)}
\end{aligned}
$$

But since we do not distinguish between horizontal-left and horizontal-right, or right diagonal up-right and right diagonal down-left, or vertical up and vertical down, or left diagonal up-left and left diagonal down-right, the functions f_1, f_2, f_3, and f_4 have additional symmetries. Assuming the spatial relationships between which we do not distinguish contribute additively, we obtain:

$$
\begin{aligned}
J(i,j) = {} & h_1(I(i,j),\ I(i,j-1)) + h_1(I(i,j),\ I(i,j+1)) && \text{(horizontal)}\\
& + h_2(I(i,j),\ I(i+1,j-1)) + h_2(I(i,j),\ I(i-1,j+1)) && \text{(right diagonal)}\\
& + h_3(I(i,j),\ I(i-1,j)) + h_3(I(i,j),\ I(i+1,j)) && \text{(vertical)}\\
& + h_4(I(i,j),\ I(i+1,j+1)) + h_4(I(i,j),\ I(i-1,j-1)) && \text{(left diagonal)}
\end{aligned}
$$

where the functions h_1, h_2, h_3, and h_4 are symmetric functions of two arguments.

Since we want the h functions to indicate relative frequency of the gray tone spatial pattern, the natural choice is to make each h the co-occurrence probability corresponding to the horizontal, right diagonal, vertical, or left diagonal spatial relationships.

This concept of textural transform can be generalized to any spatial relationship in the following way and is defined below.

Definition 2.1.3

Let P be the co-occurrence matrix for an image I as defined above. The textural transform J, $J: Z_r \times Z_c \rightarrow (-\infty,\infty)$ of image I relative to function f is defined by:

$$J(r,c) = \frac{1}{\#R(r,c)} \sum_{(a,b) \in R(r,c)} f[P(I(r,c),\ I(a,b))]$$

Assuming f to be the identity function, the meaning of J(r,c) is as follows. The set R(r,c) is the set of all those resolution cells in $Z_r \times Z_c$ in the desired spatial relation to resolution cell (r,c). For any resolution cell $(a,b) \in R(r,c)$, P(I(r,c), I(a,b)) is the relative frequency by which the gray tone I(r,c), appearing at resolution cell (r,c), and the gray tone I(a,b), appearing at resolution cell (a,b) co-occur together in the desired spatial relation on the entire image. The sum

$$\sum_{(a,b) \in R(r,c)} P(I(r,c),\ I(a,b))$$

is just the sum of the relative frequencies of gray tone co-occurrence over all resolution cells in the specified relation to resolution (r,c). The factor $\frac{1}{\#R(r,c)}$, the reciprocal of the number of resolution cells in the desired spatial relation to (r,c), is just a normalizing factor.

The transform described above is a resolution preserving transform. That is, the resolution of the output image is the same as that of the input image. While the transform image may not give rise to direct segmentation, it can be an aid in multi-spectral clustering. When Haralick (1975) included a textural transform band, he was able to improve his classification accuracy by 11 percent.

2.2 Extrema Density

Rosenfeld and Troy (1970) and more recently Mitchell, Myers and Boyne (1977) used the idea of number of extrema per unit area as a texture measure. An extrema here means a relative or local minimum or maximum gray tone.

Rosenfeld and Troy actually used extrema found in one-dimensional horizontal scans and as such were not true extrema. The idea is that the number of extrema per unit area goes up as the texture gets finer. By marking the extrema and counting the number in a square window, each pixel could be assigned a number giving a measure of extrema density. This image could then be used to detect areas with fine or coarse texture.

Another way of using extrema density for texture discrimination was by Mitchell et al. (1977). They chose some samples of textures like cork, wood, water, sand, etc. The images were smoothed and extrema in horizontal scans were computed. A threshold for the size (height) of the extrema was then selected. By varying the thresholds, the number of extrema of different sizes in each image was obtained. Plots were made for extrema size versus the number of extrema detected for that size. These gave characteristic curves for each of the texture types. The curves differed in most part between textures of different types and could be used for discrimination.

Other works using extrema have been by Mitchell and Carlton (1978) and Ehrich and Foith (1976, 1978). These will be presented in later sections.

3.0 IMAGE EXTREMA AS PRIMITIVES

The goal of our research was to study image extrema as primitives for texture analysis. We begin with some reasons as to why these were chosen as primitives and then give a formal definition for them. This is followed by a discussion on reachability sets of the extrema, which is another primitive that was examined. Finally we look at algorithms to determine the primitives in a digital image.

In the last two chapters we had some examples of primitives. From the simplest one, the pixel and its gray tone attribute, to not so intuitive ones such as gray tone runlengths and extrema pixels. The choice of a primitive is crucial as it is the basic building block of the texture model. The advantage of using a pixel lies in the fact that order is built in and spatial relationships are easy to quantize. This manifests itself in algorithms which require easy ordered scans of the image. However, this geometrical ordering is also a constraint as it restricts us from looking for primitives that were more connected with the structure of the image than the pixel. Some investigators (Peucker and Douglas, 1975; Toriwaki and Fukumura, 1978) in the last few years have been looking at image structure from a topographical point of view. This is best visualized if we represent the image three-dimensionally. Let the rows and columns be the two horizontal axes and let the gray tone value be the height. The image then takes on the perspective of mountains, valleys, plateaus, etc. A gray tone extrema then corresponds to a mountain top or the bottom of a valley. Images in this topographical view appear very complex. Figure 3.0.1 (taken from Ehrich, 1978) shows some small examples. Simple texture patterns in the original image which would easily be processed by the eye take on some very complex structures. From this point of view, it can be

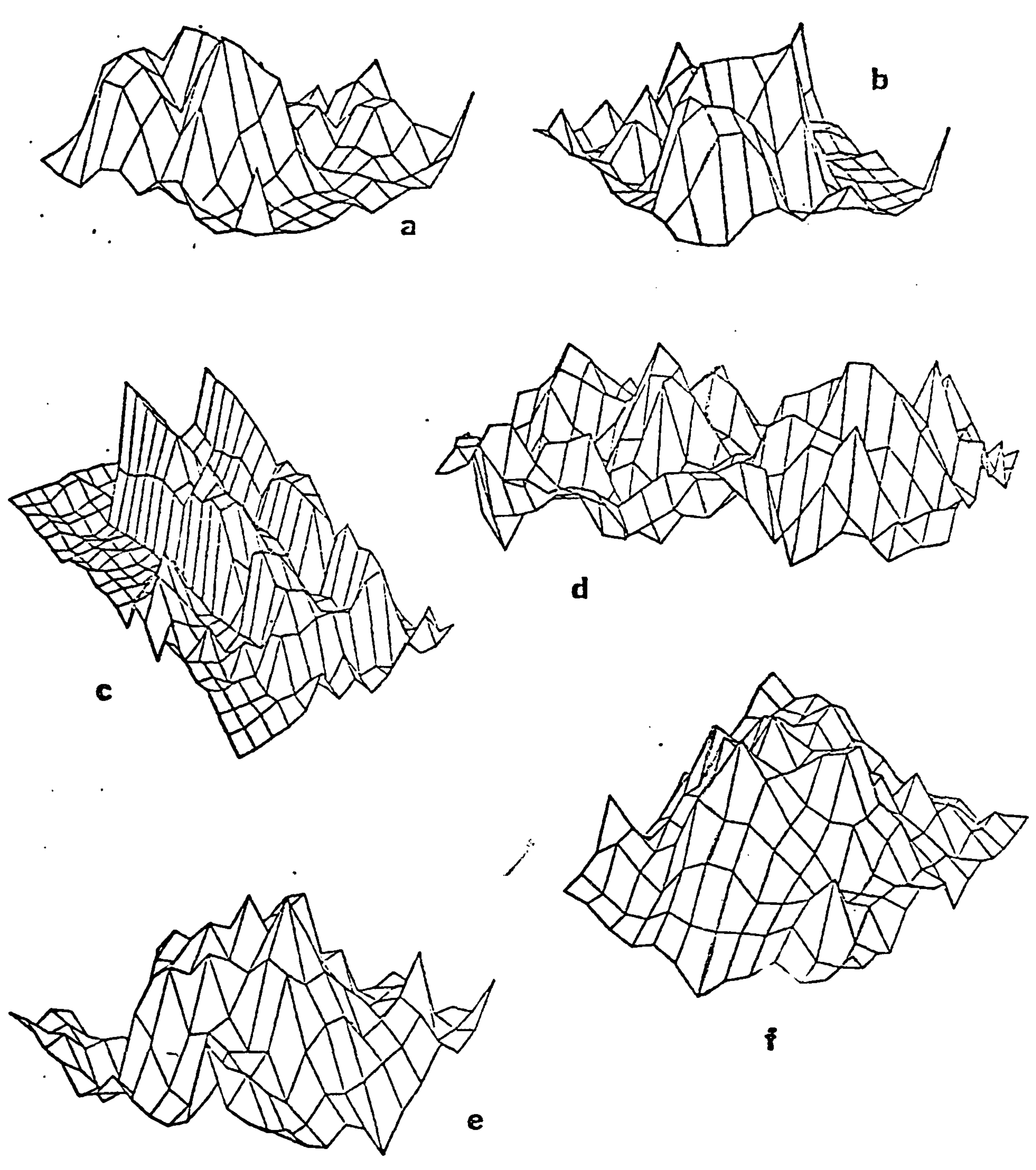


Figure 3.0.1. 3-D plots of some texture samples (from Ehrich (1978)).

seen that primitives that relate more to the basic structure of the images would be topographical entities like gray tone extrema, inflexion points, connected components. Most of these can be obtained from 3 x 3 neighborhood operators.

Toriwaki and Fukumura (1978) describe the structure (extrema, ridges, etc.) of an image by two features known as the connectivity number and radius of curvature of pixels. This link between connectivity properties of an image and extrema strengthens the hypothesis that texture is intrinsically tied to image structure.

Apart from the topographical structure, there is another good reason to use extrema as primitives. They have the property of being invariant under monotonic transformations of the gray tones. A monotonic decreasing function F on an interval I means for all x, y $\in$ I: $x \leq y \rightarrow F(x) \leq F(y)$. This means that if we put the gray tones through a monotonic transformation, our extrema locations will be unchanged. This proves useful, for example, in extracting texture from objects, parts of which lie in a shaded area.

Another view which connects extrema to the structure of an image is provided by an automata defined by an image. There are many different ways to do this and the definition given below uses the four neighborhood (vertical and horizontal neighbors) of a pixel. The transition function is defined by the relationships between gray tones of the pixel and its neighbors. The states are the pixels of the image.

Definition 3.0.1

An automaton, A, defined by the image I is the triple $(Z_r \times Z_c, \Sigma, \delta)$, where the input alphabet Σ is the set {N,S,E,W} and the transition function δ is the mapping $\delta:(Z_r \times Z_c) \times \Sigma \rightarrow Z_r \times Z_c$. It is defined as:

$$\delta((i,j),N) = (i - 1, j), \text{ if } I(i,j) \geq I(i - 1, j)$$
$$= (i,j) \text{ otherwise}$$

$$\delta((i,j),S) = (i + 1, j), \text{ if } I(i,j) \geq I(i - 1, j)$$
$$= (i,j) \text{ otherwise}$$

$$\delta((i,j),E) = (i, j + 1), \text{ if } I(i,j) \geq I(i, j + 1)$$
$$= (i,j) \text{ otherwise}$$

$$\delta((i,j),W) = (i, j - 1), \text{ if } I(i,j) \geq I(i, j - 1)$$
$$= (i,j) \text{ otherwise}$$

In automata terminology, the maxima pixels correspond to inaccessible states or generators ɔf primaries. The minima pixels become the set of strongly connected subautomata. Later on we will look at the reachability sets of the extrema. These are called the descending and ascending components of the maxima and minima, respectively. The descending components correspond to states that occur in one primary only while the ascending components map to states that occur only once in the set of sources of the strongly connected subautomata (Bavel, 1968).

The automata is a very strong mathematical tool. The analogies between the extrema of an image to some of the basic features of an automata, provides further evidence to their worth. As we shall see later, this relationship between an image and its automata is exploited to define the reachability sets of the extrema.

3.1 Definitions

Having introduced our primitives, we would like to formalize them. This is done through a series of definitions of connectedness properties and neighborhoods in an image. We begin with the four and eight neighborhoods of a pixel.

Definition 3.1.1

Let $Z_r \times Z_c$ be a set of resolution cells. The 4-neighborhood (N_4) of a pixel (i,j) is the set of cells:

$$N_4(i,j) = \left\{(m,n) \in Z_r \times Z_c \mid \text{either } m = j \text{ and } j - 1 \le n \le j + 1 \text{ or } n = j \text{ and } i - 1 \le m \le i + 1\right\}$$

The 8-neighborhood (N_8) of a pixel (i,j) is the set:

$$N_8(i,j) = \left\{(m,n) \in Z_r \times Z_c \mid i - 1 \le m \le i + 1 \text{ and } j - 1 \le n \le j + 1\right\}$$

Next we look at a connected sequence of cells and equality paths.

Definition 3.1.2

Let $S = <(m_0,n_0,(m_1,n_1),\ldots,(m_K,n_K)>$ be a sequence of cells in $Z_r \times Z_c$. S is said to be a 4(8)-connected sequence if and only if:

$$(m_i,n_i) \in N_{4(8)}\,(m_{i-1}, n_{i-1}),\ i = 1,2,\ldots,K$$

Definition 3.1.3

Let S be a connected sequence of cells as above. S is called an equality path if and only if:

$$I(m_i,n_i) = I(m_{i-1}, n_{i-1}),\ i = 1,2,\ldots,K$$

An equality path is a connected sequence in which all pixels have the same gray tone. No neighborhood was specified above. Unless otherwise stated, we will assume an eight neighborhood, N_8, or N from now on.

An equality path allows us to define the flat of a pixel.

Definition 3.1.4

The flat of a pixel (r,c), denoted by F(r,c) is set of pixels:

$$F(r,c) = \left\{ (m,n) \mid \text{there exists an equality path from } (r,c) \text{ to } (m,n) \right\}$$

The flat of a pixel may be the pixel itself. All pixels have flats and these are maximal in size as given by the definition above. A flat is then a maximally connected region with the same gray level.

Given the flat of a pixel we can talk about the boundary pixels in the flat. A boundary pixel of a flat is one which has at least one neighbor which is not part of the flat.

Definition 3.1.5

Let F(r,c) be the flat for pixel (r,c). The set of boundary pixels for the flat denoted by $F_B(r,c)$ is the set:

$$F_B(r,c) = \left\{ (m,n) \in F(r,c) \mid \exists (p,q) \in N(m,n) \text{ and } (p,q) \notin F(r,c) \right\}$$

The boundary pixels of a flat allow us to determine if the flat is a relative extremum or not.

Definition 3.1.6

Let (r,c) be a pixel in $Z_r \times Z_c$. F(r,c) is a relative maxima (minima) if and only if for all $(p,q) \in F_B(r,c)$:

$$I(p,q) \geq (\leq)\ I(m,n), \forall (m,n) \in N(p,q)$$

The extrema or relative extrema are usually single pixels. However, if the flat consists of more than one pixel, we call the entire set an extremum of the image.

3.2 Reachability Sets

In this section we introduce an extension of the image extrema. These are called the reachability sets or descending/ascending components of the maxima/minima. Essentially the descending component of a maxima is all the cells that can be reached in non-increasing gray tone paths. Correspondingly, the ascending component of a minima are all the cells that can be reached in non-decreasing gray tone paths. In the three-dimensional perspective of the image of Section 3.0, this amounts to traveling down from the mountain tops or going up from the valleys. The computation of the ascending/descending components, reachability sets, or transitive closures actually is a spatial clustering on the image. It is in this framework that the following discussion is presented. Later we will amend it to suit our needs, i.e., to get reachability sets which will themselves be used as primitives.

The discussion below restricts itself to descending components of maxima as they are intuitively easier to comprehend. A similar argument can be presented for ascending components of minima with the definition of the relation R below adjusted. The next two sections look at the transitive closure of a cell and how to compute it.

3.2.1 Definitions

Spatial clustering can be thought of as grouping together units which bear a similarity to each other and in addition have some spatial relationship. For most cases, the formal definition of the cluster and the algorithm which generates it go together. In this and the next few sections, we look at this description.

We begin with the definition of the city block or Diamond distance function.

Definition 3.2.1

Let ρ be a digital distance function defined on digital image. ρ is called the Diamond or City Block distance function if:

$$\rho((i,j),\ (k,m)) = |i - k| + |j - m| \qquad (i,j),\ (k,m) \in Z_r \times Z_c$$

In most of what follows, we use this distance function to illustrate the algorithm. As is discussed later, the algorithm is not restricted by this distance function. It just happens to be one that is easy to visualize.

To characterize the clusters we define a binary relation $R \subseteq (Z_r \times Z_c) \times (Z_r \times Z_c)$ by:

$$R = \left\{((i,j),\ (k,m))\ \middle|\ I(i,j) \geq I(k,m),\ \rho((i,j),\ (k,m)) \leq \theta\right\}$$

where

$(i,j),\ (k,m) \in Z_r \times Z_c$

ρ = the digital distance function described above, defined on $Z_r \times Z_c$

For now we take $\theta = 1$. The relation R then takes on a meaning that is readily discernable. For $\theta = 1$, R consists of ordered pairs of cells such that going from the first to the second cell takes one step and the gray levels are non-increasing. This description may be viewed as a path of length 1 from the first to the second cell. $\theta < 1$ or $\theta = 0$ implies a path of length zero, e.g., the element $((i,j),\ (i,j)) \in R$. In general, a path of length n between two cells can be defined as below.

Definition 3.2.2

Let (x_0,y_0), $(x_n,y_n) \in Z_r \times Z_c$. Let $<(x_0,y_0), (x_1,y_1)\ldots(x_n,y_n)>$ be a sequence of cells such that for all $i \in \{1,2,\ldots,n\}$, $((x_{i-1}, y_{i-1}), (x_i,y_i)) \in R$. Then the sequence $<(x_0,y_0),(x_1,y_1)\ldots(x_n,y_n)>$ is called a path of length n from (x_0,y_0) to (x_n,y_n).

The definition of a path here is different from the equality paths of the last section. The modification here is with regard to the non-increasing of gray levels as we proceed down the sequence. This will be discussed further later.

Definition 3.2.3

The transitive closure of the relation R, denoted by R^T, is defined as:

$$R^T = \bigcup_{i=1}^{\infty} R^i, \text{ where } R^i = \underbrace{R \circ R \circ \ldots \circ R}_{i \text{ times}}$$

We have seen R consists of ordered pairs of cells which are end points of paths of length 1. $R \circ R$ then consists of ordered pairs of cells of $(Z_r \times Z_c) \times (Z_r \times Z_c)$ which are end points of paths of length 2. Thus, $R \circ R \circ \ldots \circ R$ (i times) consists of ordered pairs of cells which are end points of paths of length i. The union of all these sets defines the transitive closure of R.

We are interested in the transitive closure or reachability set of a cell. This is defined in an analogous manner to R^T above.

Let $R(m,n) = \{(i,j) \mid ((m,n), (i,j) \in R\}$. We note that the City Block digital distance function as used in the definition of R for $\theta = 1$ gives for R(m,n) the four-neighborhood of (m,n) (the east, west, north, south neighbors of (m,n)), if they are reachable, along with the cell (m,n).

Definition 3.2.4

The transitive closure of the cell $(m,n) \in Z_r \times Z_c$ is defined as:

$$R^T(m,n) = \bigcup_{i=1}^{\infty} R^i(m,n) \text{ where}$$

$$R^i(m,n) = \{(i,j) \mid ((m,n), (i,j)) \in R^i\} \quad (i,j) \in Z_r \times Z_c$$

Thus, $R^T(m,n)$ is a sub-image of $Z_r \times Z_c$. $R^1(m,n) = R(m,n)$ consists of (m,n) and those cells which can be reached from (m,n) in one step, i.e., there is a path of length 1 from (m,n) to those cells. $R^2(m,n)$ consists of (m,n) and those cells which can be reached from (m,n) in two steps, or there is a path of length 2 from (m,n) to those cells. In general $R^i(m,n)$ consists of (m,n) and those cells reachable from (m,n) in i steps or those cells which lie a path length i away from (m,n). So $R^T(m,n)$ consists of all cells reachable from (m,n) or those cells such that a path exists from (m,n) to them.

The clusters we wish to determine in an image are defined in terms of $R^T(m,n)$. Given a cell (m,n) we wish to determine the closure of (m,n). This may be called a cluster. However, we want these clusters to be maximal in size, thus generating from the image maxima. Borrowing a term from automata theory we will call these maximal clusters primaries. They are defined formally as follows.

Definition 3.2.5

$V \subseteq Z_r \times Z_c$ is a primary of $Z_r \times Z_c$ if and only if:

(1) $V = R^T(m,n)$ for some $(m,n) \in Z_r \times Z_c$, and

(2) If $(i,j) \in Z_r \times Z_c$ and $V \subseteq R^T(i,j)$, then $V = R^T(i,j)$

3.3 Algorithm to Compute Transitive Closure of a Cell

First we present the theoretical aspects of the algorithm. To do this we introduce some notations:

$$\text{Let } \alpha(i,j) = \min_k \{k \mid I(i,k) \leq I(i, k+1) \leq \ldots \leq I(i,j)\}$$

$$\beta(i,j) = \max_k \{k \mid I(i,j) \geq I(i, j+1) \geq \ldots \geq I(i,k)\}$$

$\alpha(i,j)$ and $\beta(i,j)$ are then the column coordinates of the last cell reachable, when traveling horizontally left and right, respectively, from (i,j). Similarly, let

$$\gamma(i,j) = \min_k \{k \mid I(k,j) \leq I(k+1, j) \leq \ldots \leq I(i,j)\}$$

and

$$\delta(i,j) = \max_k \{k \mid I(i,j) \geq I(i+1, j) \geq \ldots \geq I(k,j)\}$$

$\gamma(i,j)$ and $\delta(i,j)$ are row coordinates of the last cell reachable when traveling vertically up and down, respectively, from (i,j).

From the definitions above, the following always hold:

$$\left.\begin{array}{l} \alpha(i,j) \leq j \\ \beta(i,j) \geq j \\ \gamma(i,j) \leq i \\ \delta(i,j) \geq i \end{array}\right\} \quad -(1)$$

In terms of these end row and column pointers we get two sequences of cells. All cells reachable from (i,j) traveling horizontally or vertically, i.e., the sets:

$$H(i,j) = \{(i,n) | \alpha(i,j) \leq n \leq \beta(i,j)\}$$

and

$$V(i,j) = \{(m,n) | \gamma(i,j) \leq m \leq \delta(i,j)\}$$

From the definition of these two sets and (1) above, it follows:

$$\left.\begin{array}{l}(i,j) \in H(i,j) \text{ and} \\ (i,j) \in V(i,j) \text{ always}\end{array}\right\} \quad -(2)$$

Thus, $H(i,j)$ is the set of all points which can be reached from (i,j) going horizontally in both left (west) and right (east) directions. $V(i,j)$ is a similar set for the vertical paths from (i,j) going up (north) and down (south).

The algorithm consists of finding all these vertical and horizontal sequences in an iterative manner. For this purpose we define a sequence of sets $A_1, A_2, \ldots, A_n$ as follows:

$A_1 = \{(p,q)\}$ where $(p,q) \in Z_r \times Z_c$ is the cell whose closure we wish to compute and

$$A_{n+1} = \bigcup_{(i,j) \in A_n} H(i,j)$$

$$A_{n+2} = \bigcup_{(i,j) \in A_{n+1}} V(i,j)$$

Thus, by the above definitions and (2) it follows that $A_i \subsetneq A_{i+1}$, for all positive integers i.

Before we prove a Lemma relating the sequence of sets $A_1, A_2, \ldots, A_n$ to $R^T(p,q)$, a special note should be made regarding paths in a digital image as defined in Defintion 3.2.2. Under the constraints of the form of ρ (City Block distance function) and θ ($\theta \leq 1$), a path between two cells in $Z_r \times Z_c$, consists of alternating horizontal and vertical subpaths only. This is because the definition of ρ and θ, allow at most only the two vertical and two horizontal neighbors of a cell (m,n) in R(m,n). Thus, any cell in a path in $Z_r \times Z_c$ is in either a horizontal or a vertical relationship to its predecessor (we are using the four-neighborhood here). We now prove the above-mentioned Lemma.

Lemma 3.3.1

$A_\infty = R^T(p,q)$

Proof

Let $(i,j) \in R^T(p,q)$.

Then there exists a path from (p,q) to (i,j) but for any resolution cell to be in such a path, it must be in either a horizontal or vertical relationship to its predecessor.

Thus, the path can be broken up into alternating horizontal and vertical oriented subpaths. The first horizontally oriented subpath gets picked up in A_2. The first vertically oriented subpath gets picked up in A_3. Eventually all the subpaths are included in A_∞. Therefore, $(i,j) \in A_\infty$ and so $R^T(p,q) \subseteq A_\infty$.

Let $(i,j) \in A_\infty$. By definition of A_∞, it must include all resolution cells on all paths made up of alternating horizontal and vertical oriented subpaths of $R^T(p,q)$. Thus, $(i,j) \in R^T(p,q)$ and so $A_\infty \subseteq R^T(p,q)$. Thus, $A_\infty = R^T(p,q)$.

The structure of the algorithm is as follows. Start with the cell (p,q) the set A_1 whose closure we wish to get. In the first generate A_2, or all cells which can be reached going horizontally from (p,q).

In the next scan generate A_3 or all cells which can be reached from each cell in A_2, traveling vertically. Continue scanning generating $A_4, A_5, \ldots$ This will give us $R^T(p,q)$.

By the Lemma, we know that we will get $R^T(p,q)$ in an infinte number of scans. However, that is not practical or for that matter, not even algorithmic. In practice, for images of finite size, the number of scans necessary, is finite. We let the procedure terminate when no new cells were added in the last scan. It remains to show that this will always happen and hence we will always terminate in a finite number of steps. Also, when we do stop we have got exactly $R^T(p,q)$. The following Lemma shows this.

Lemma 3.3.2

Let $A_1, A_2, \ldots, A_n$ be a sequence of sets as defined above. Then:

(1) There exists a positive integer n such that

$$A_n = A_{n+1} = \ldots = A_\infty$$

(2) $A_n = R^T(p,q)$

Proof

Since $A_i \subseteq A_{i+1}$, for any positive integer i, the total number of cells in A_i, for each scan cannot decrease.

If the number of cells absorbed is always increasing in successive scans, then eventually the whole image will be exhausted, since it is of finite size. Further scans cannot add anything new, thus part (1) holds and we terminate the procedure. (This is the case when the whole image belongs to $R^T(p,q)$).

Suppose then for some scan n, $A_{n+1} = A_n$. There were no new cells added. Either A_{n+1} was a horizontal scan or a vertical scan. (Horizontal scan means a scan in which we were looking for horizontal paths).

Let A_{n+1} be a horizontal scan. Then A_{n+1} implies there were no new horizontal paths added in the $(n + 1)^{th}$ scan. Thus, the $(n + 2)^{th}$ scan, (a vertical scan) cannot absorb any new members, since no new cells were obtained in the previous horizontal scan, which would have generated new vertical paths. And if there were any vertical paths from the cells in A_{n+1}, they would have been picked up in the n^{th} scan. A similar argument holds for the next (horizontal) $(n + 3)^{th}$ scan. No new members can be generated.

If A_{n+1} were a vertical scan, a similar discussion, with the words horizontal and vertical interchanged, would give that successive scans would yield the same sets.

Thus, $A_n = A_{n+1}$ implies $A_n = A_{n+1} = A_{n+2} = A_{n+3} = \ldots A_\infty$ where n is a positive integer.

From Lemma 3.3.1 we have $R^T(p,q) = A_\infty$ and by part 1 of this lemma $A_n = A_\infty$. So $R^T(p,q) = A_n$.

The two lemmas thus constitute a proof for the algorithm.

In the implementation of the algorithm one has to keep track of some details like which cells have been included already or which cells should the paths be checked from. It is quite redundant to generate paths from cells more than once, as we would gain nothing new. These details may be kept track of by using two markers. Thus, if the horizontal path from a cell has been generated in a scan, it should be marked "horizontally-done." It may thus be skipped over in later horizontal scans. These markers can then be used in generating path from only new cells gotten in the previous scan. For example, new vertical paths are generated from those cells which are marked "horizontally done" by the previous scan, but not "vertically done."

At the end of the generation process those cells marked vertically and horizontally done will belong to $R^T(p,q)$. Some examples will make this clearer.

We present two examples with discussion as to how the algorithm acts on them. They have been simplified to enable better visual understanding. Only one primary per picture is discussed (usually there are many more) and we ignore gray tones, assuming that the shaded areas are the primaries (closure) of mountain tops at X.

In Figure 3.3.1 we have a simple blob primary as marked out. The generator is X. Here $A_1 = \{X\}$. In the first scan we will generate the horizontal path (A_2 = {set of cells between A and B}) from X, marking every cell in this path with, say "-", as "horizontally done." In the second scan we will generate vertical paths only from cells marked with an "-". All these cells (i.e., set A_3) in the vertical paths will be marked "|" to denote "vertically done." At this stage, cells on the line AB will be marked "+" as both horizontal and vertical paths have been generated from them. A_3 then will consist of all points in the shaded area except cells C and D as they have not been reached yet. The third scan (horizontal) will generate A_4 which shall include C and D. From this scan we will generate horizontal paths from cells marked with "|" only. Two of these paths will include C and D. At the end of this scan all cells in the primary would have been gotten and all except C and D will be marked "+". C and D will contain only "-" marks, and will need

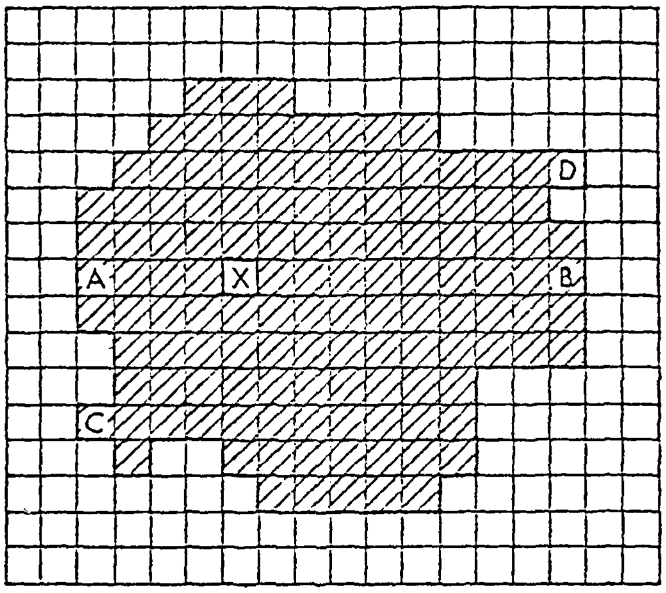


Figure 3.3.1. Example of primary generation.

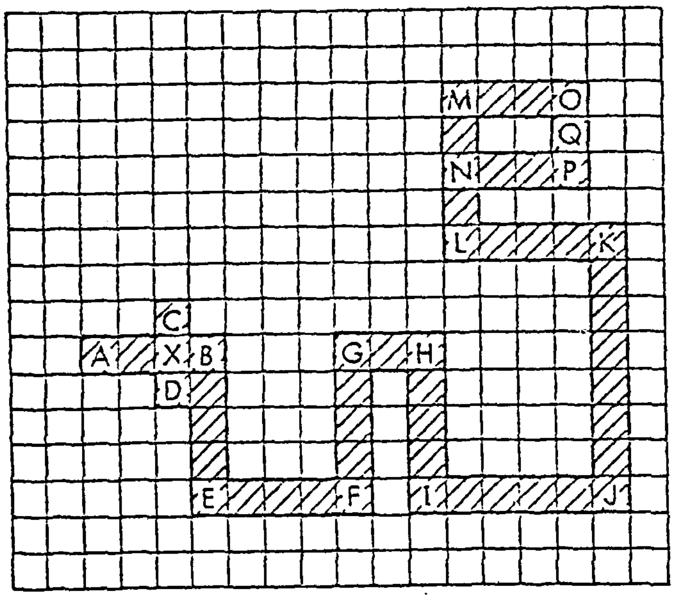


Figure 3.3.2. Example of primary generation.

to have their vertical neighbors checked for paths in the next (fourth) scan. This, of course, will not give us any new cells and C and D will be marked "+". The whole primary has been generated and the process will stop here, $A_4 = A_5$.

What determined the value n = 5 here? We note, if it was not for C and D there would have been one less scan. We need at least three scans here since starting with the horizontal direction, three alternating (horizontal-vertical-horizontal) subpaths are needed to reach C and D. The fourth scan just determines that we have all the cells.

The number of scans needed is the minimum number of alternating subpaths needed (plus one) such that all the cells are reached. This becomes evident on examining Figure 3.3.2. It looks quite pathological but serves to illustrate the problem.

Here we have the generator of the primary, X. ($A_1 = \{X\}$). In the first (horizontal) scan we will get A_2 = {cells on the line AB}. The second (vertical) scan will yield in addition to A_2, $A_3 = A_2 \cup$ {cells C and D and cells on vertical line BE}. The next horizontal scan will yield in addition to what we have gotten the cells on line EF. Thus, the pattern is as follows. Here we let {RS} denote cells lying betweeen and including the cells R and S:

$A_1 = \{X\}$

$A_2 = A_1 \cup \{AB\} = \{AB\}$

$A_3 = A_2 \cup \{C,D\} \cup \{BE\}$

$A_4 = A_4 \cup \{EF\}$

$A_5 = A_4 \cup \{FG\}$

$A_6 = A_5 \cup \{GH\}$

$A_7 = A_6 \cup \{HI\}$

$A_8 = A_7 \cup \{IJ\}$

$A_9 = A_8 \cup \{JK\}$

$A_{10} = A_9 \cup \{KL\}$

$A_{11} = A_{10} \cup \{LM\}$

$A_{12} = A_{11} \cup \{NP\} \cup \{MO\}$

$A_{13} = A_{12} \cup \{Q\}$

$A_{14} = A_{13}$ and the procedure stops.

Here we needed 13 scans since a minimum of twelve alternating subpaths are needed to reach Q from X.

This may seem like quite a lot, but in most other methods to generate transitive closure usually more than 13 scans would be needed. The reason being most other methods look for paths in only one direction per scan. This algorithm looks at two directions in a scan (either left-right or up-down), which gives it more power in the general case.

3.4 Algorithms

In this section we will look at some more algorithms. The first two of these compute the extrema of the image. Then follows an algorithm to give each extrema a unique label. The last is another algorithm for reachability sets. The algorithm of Section 3.3 computed the descending component of one maximum only. In general, the descending components of different maxima overlap and thus if the entire component is desired, each has to be computed separately. We use a modified version of the reachability set of an extremum. This set consists of those pixels which are reachable by the extremum and no other. In otherwords, if a pixel is reachable by two maxima (or minima) it is considered in a special overlap region and ignored. These sets are called the unique reachability sets. Thus, the result of the last algorithm on the image containing the maxima, is another image with the maxima grown out over their unique descending components. By definition the ascending components of the maxima are the maxima themselves. A similar discussion holds for the minima.

In order to maintain consistency and allow for comparison, the examples presented are the result of applying the various algorithms to one image, designated as M4A. This is a 128 x 128 section of the fourth image provided by ETL. The "A" stands for the first subsection examined from this image. The two bands are shown in Figure 3.4.1. The first is the radar image while the second is the aerial photo image. The section is of a mostly residential area with a trailer court at the bottom left and a plain field at the top left. It was chosen for its three sample textures. For most of the processing the aerial photo was used as it had less noise.

The usage and documentation of the algorithms is described in the Appendix. Basically they have all been set up as image operators. That is, all the subroutines and procedures for each algorithm were put together in a mini-package. This resulted in an operator or command which was applied to the image. Each operator takes in one or two images as input

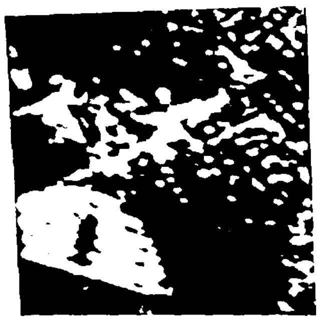


Figure 3.4.1a. Band 1 of M4A - Radar Image. Image size 128x128.

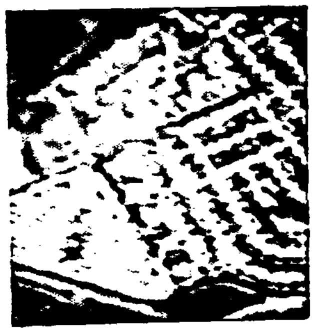


Figure 3.4.1b. Band 2 of M4A - Aerial Photo. Image size 128x128.

and generates a resulting output image. This output image can be operated upon by other commands as necessary.

There are four commands - MRKNX, MNMX8, LBLCT, REACH. The first two compute the extrema of the image. LBLCT is used to label each extrema region uniquely, while REACH computes the unique reachability regions.

The first, MRKNX, stands for mark local minima and maxima pixels. It takes in a gray tone image and outputs an image in which each pixel is marked as a minima, maxima, flat or transition, as determined by applying a 3 x 3 window over each pixel. It is a one pass operation. The output values do not represent true relative extrema as only a 3 x 3 window for each pixel is examined. The pixel is marked 1 or 2 if it was the minimum or maximum of the nine cells (itself and its eight neighbors). It is given a value 0 (flat) if all nine pixels had the same value. Finally, if it is none of the above, it is marked as transitionary and given a value 3.

In the example below, Figure 3.4.2, the center cell will receive the various labels.

3	5	2
4	1	7
6	8	4

Minimum = 1

2	2	1
2	5	3
5	4	3

Maximum = 2

5	5	5
5	5	5
5	5	5

Flat = 0

2	5	5
2	3	4
2	3	4

Transitionary = 3

Figure 3.4.2. Labels assigned to the center pixel by MRKNX.

The MRKNX operation results in an image which serves as "seeds" for a region growth process by the iterative MNMX8 operator.

The commands MNMX8, LBLCT, and REACH, while they do different things, all have the same basic structure and scan the image by applying a 2 x 2 window template. The effect of these programs is to apply a rule to the cell pairs defined by the template on the image, until all these pairs satisfy some relation. The scan increment is one cell at a time and the 2 x 2 window covers all adjacent cell pairs as is illustrated below. Figure 3.4.3 shows the scan template. The arrows indicate which pairs of cells are compared. The template and scanning work on the eight neighborhood of a pixel.

There are four comparisons made for each window positioning. When the template is passed across the image in one pixel increments, all of the adjacent (eight neighbors) of every cell in the image are covered during each scan of the image. This can be seen in Figure 3.4.4. Thus, the center pixel is compared to each of its neighbors. For MNMX8 and REACH the window is moved from left to right and right to left as well as top to bottom and bottom to top. For LBLCT the scan goes horizontally and from top to bottom. Each scan results in propagation of labels, markers, etc. The scanning iterations continue until no change is recorded. This results in the output image for the command.

MNMX8 is a recursive filter which uses the original gray tone image and the output of MRKNX to label extrema. Labels (0,1,2,3) in the image from MRKNX are propagated till all flats are eliminated. The propagation rule on a pair of cells for a few labels of the marked image is given below.

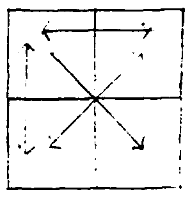


Figure 3.4.3. Scan template for MNMX8, LBLCT, and REACH operators. Four cell-pair comparisons are made for each template positioning.

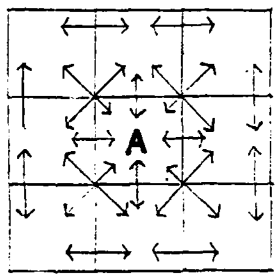


Figure 3.4.4. All eight neighbors for pixel A are covered once by moving the template across one column and down one row.

Propagation need only be performed if the two labels are not the same and the corresponding two gray tones are equal.

Let the two cells be x and y with the marked labels L_x and L_y. We have three cases to examine for when $L_x \neq L_y$ and the gray tones of x & y are the same:

(a) Either L_x or L_y equal 0 and the other non-zero (1,2,3). Output is the non-zero label propagated into the 0 label cell.

(b) Either L_x = 1(min) and L_y = 2(max) or vice versa. Output in both cells get marked 3 (transition) as a region cannot be both a minimum and maximum at the same time.

(c) Either L_x = 3 or L_y = 3. The output is 3 for both cells regardless of the other value. The transition label is propagated since if a region is known to have a transition label, all its cells must be marked transition also.

The image is iterated with the scanning template till no more propagation is possible. The propagation must cease as in each iteration the number of pixels marked 0, 1, or 2 never increase while the number of pixels marked 3 never decrease. Unless the image is pathological (i.e., all pixels have the same value), there will be at least one maxima and one minima. The output image contains no zeros. The cells marked 1 and 2 represent the true relative minima and maxima.

Figure 3.4.5 show these for M4A. The second photo image was used to determine both the local 3 by 3 mark labels as well as for the propagation of these labels. The minima are red and the maxima green, though on printing the latter appear black.

In order to use either the minima or maxima, each one has to be labeled uniquely. This allows for both property generation as well as the generation of the reachability sets. The LBLCT command does this for us. If the maxima were specified, then it examines each maximally connected set

Figure 3.4.5a. Local (3x3 window) maxima (green), minima (red), transitionary (yellow) and flat (black) pixels. Result of applying the MRKNX operator. (Shown at a larger scale.)

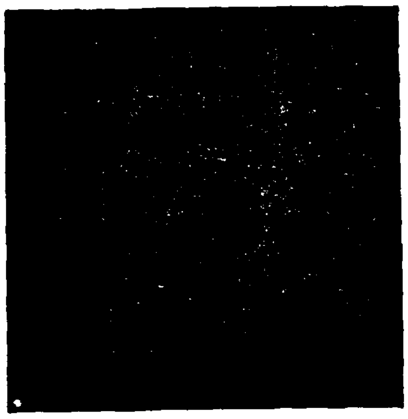


Figure 3.4.5b. Relative minima (red) and maxima (green). Result of applying the MNMX8 operator. (Shown at a larger scale.)

of pixels marked 2 and assigns unique labels to each set. The pixels not marked 2 are ignored or treated as background markers.

LBLCT achieves its labeling by using the scanning template described previously. It scans horizontally and goes from top to bottom of the image. It also uses a linked list and counters to keep track of the number of regions encountered. The propagation function for the labels of a pair of cells in the output image is described below.

Let the two pixels be x and y. L_x and L_y are the labels in the output image. M_y is the mark value of cell y in the input (result of MNMX8) image. The discussion below is for labeling maxima, i.e., cells for which mark value is 2.

Initially the labeled image is all zero. A counter which maintains the current label count is initialized to 1. A linked list for the labels is set up. Each entry is initialized to point to itself, e.g., list(5) = 5 states that pixels that have label 5 are connected to region number 5. At a later stage this may change to say list(5) = 3. This means that pixel labeled 5 should have the same label as those for region number 3, as pixels with labels 3 and 5 were once assumed as separated but turned out to belong to the same region. At the end of scanning the image, the smallest linked label from the list is chosen and the output image is relabeled to generate the unique labels.

If cell y has a mark value of 2, it gets labeled one of three ways in the output image:

(a) If $L_y \neq 0$, L_y is set to the label that is linked to the linked list. It is possible that the result of the relabeling will produce the same result as before, as all labels start out being linked to themselves. After this relabeling L_x is examined. If $L_x \neq 0$ then list $(\text{Max}(L_x,L_y) = \text{Min}(L_x,L_y)$. The larger label is linked to the smaller one. If $L_x = 0$ no action is performed.

(b) If $L_y = 0$ then if $L_x \neq 0$, $L_y = L_x$.

(c) Case for both $L_x = L_y = 0$. L_y then gets set to the next new label.

Figure 3.4.6 shows the uniquely labeled extrema for M4A.

The REACH operator generates reachability set for either the minima or maxima. Input is the gray tone image and a uniquely labeled extrema image. The cells not belonging to any maxima have been given a special label Ø for background. The scanning is done as before. The propagation rule is applied if the label of the two cells are not the same. The rule for descending components is as follows. Let the gray tones and labels of the first and second cells be x, y, and L_x, L_y.

(a) If $(x < y)$ then no propagation

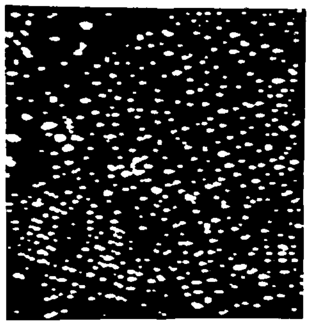
(b) If $(L_x = 0)$ then no propagation

(c) If $(L_y = 0)$ then $L_y = L_x$; otherwise $L_y = \text{OVL}$

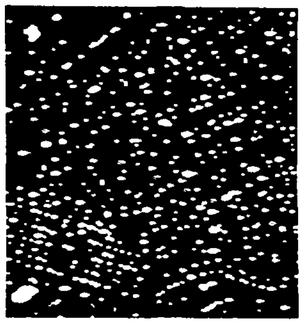
OVL is the overlap label which indicates that a pixel is reachable from two maxima. A similar rule exists for the ascending components.

Figure 3.4.7 shows the reachability sets for the extrema of M4A.

Next we look at some processing with these and other algorithms.

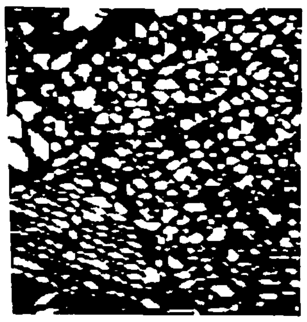


(a) Maxima

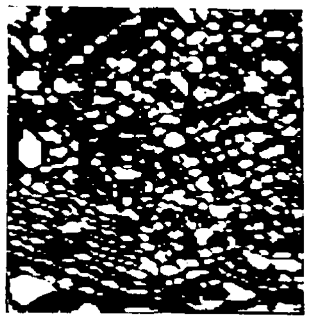


(b) Minima

Figure 3.4.6. Uniquely labelled Maxima and Minima of M4A. Result of applying the LBLCT operator to the maxima and minima pixels respectively of Figure 3.4.5b.



(a) Descending Components



(b) Ascending Components

Figure 3.4.7. Descending components of the Maxima and Ascending components of the Minima pixels. These are the unique reachability sets of M4A, as a result of applying the REACH operator. The overlap region is black.

4.0 EXPERIMENTS WITH EXTREMA DENSITY

This chapter discusses experiments carried out with extrema density. Essentially this is a measure of number of extrema per unit area. Experiments were done using either maxima, minima, or both. Density images were also generated from the reachability sets.

In Chapter 2.0 we mentioned two investigations which used extrema density for texture analysis. Their extrema were computed along horizontal scan lines only and were not true extrema. A more recent work which achieves image segmentation using extrema is by Mitchell and Carlton (1978). In addition to using just frequency of extrema, they also make use the height attributes of extrema, which we discuss in the next chapter. Their extrema are computed by combining horizontal and vertical one-dimensional scan operations. An extremum in one-dimension is found if the gray tones rise or fall beyond preset levels (heights). Thus, these extrema also differ from the relative min/max presented in the last chapter.

Images are generated in which each pixel is given a count of the number of extrema in a 60 x 60 window surrounding it. Different images with different thresholds of extrema height are generated. These are then used together in a multi-spectral clustering algorithm for segmentation. Hierarchical segmentation is achieved by using smaller window sizes and segmenting within the large regions obtained in previous iterations. They achieved fairly good results with this method.

As in the last chapter, all images presented here are the result of processing image M4A of Figure 3.4.1 to different degrees. A large number of images were generated and examined. In the next few sections we summarize the experiments and results.

4.1 Extrema Density

The problem with simply counting all the extrema in the same extrema plateau as extrema is that extrema per unit area is not sensitive to the difference between a region having few large plateaus of extrema or many simple pixel extrema. One solution to this is to count each extrema plateau once. This involves locating some central pixel in the extrema and marking it as the extrema associated with the plateau. The problem was solved in the experiments performed by taking a weight "W" and assigning a value W/N for each pixel in the N-celled extrema. Thus, if the weight was 100, each single celled extrema would get a value 100; each cell in a two-cell extrema would get a value 50; the three cells in a three-cell extrema would get values 33, 33, and 34; and so on. To achieve this the size of each extrema region had to be determined. The algorithm for that is discussed in the next chapter under extrema attributes.

The choice of the weight is arbitrary as long as it is larger than the size of the largest extrema. For our experiments a weight of 255 was chosen for no other reason than the fact that no more than 8 bits of significance would be required to store image values. Higher weights would have resulted in larger number of bits and would have used more disc space.

In order to obtain density images, the image with weight-distributed extrema was filtered repeatedly. The filters applied were averaging box filters, of window sizes 3 x 3 and 5 x 5.

The result of applying a 3 x 3 box filter once is to replace the gray tone of a pixel by the average of the gray levels of its neighbors and itself. This is a low pass filter operation and results in a smooth image or defocused image.

The filter can be represented as a window as shown in Figure 4.1.1. Each cell has a weight of 1, and the resulting sum for a pixel is divided by the sum of the weights, 9. One iteration does not do much defocusing. Either we can use a larger window like 10 x 10, or 20 x 20, or apply a small window repeatedly. Using a large window has a drawback. The resulting image is streaked. This is a consequence of giving all the cells in the large window the same weight. It makes more sense to weigh the center cell more and reduce the weight as we go towards the edges of the window. This is exactly what results in apply a smaller filter repeatedly. The rest of Figure 4.1.1 shows the distribution of weights when a 3 x 3 window is applied up to 5 times. The effect of applying two 3 x 3 filters is to apply the filter "332" once. The size of filter is larger but the weighting is no longer uniform. Figure 4.1.2 shows the corresponding filters for repeated 5 x 5 iterations.

Let us examine this in the general case. Let the size of the filter be S x S. We will assume S to be odd and can represent it as S = 2m + 1, m is an integer. If we apply this filter n times, the resulting filter will be (2nm + 1) x (2nm + 1) in size, and the total weight associated with the filter is S^{2n}.

In the next set of figures we have the results. Figure 4.1.3 shows the weighted maxima image of M4A and the results of applying five 3 x 3 and five 5 x 5 box filter operations. The density effect starts to come about the third application of the filter. We could have gone on beyond five iterations, but did not. In these density images a brighter area corresponds to higher texture density or finer texture. The dark areas correspond to lower texture density or coarser texture. These should be compared to the original M4A photo image of Figure 3.4.1, for texture

1	1	1
1	1	1
1	1	1

Filter 331. Total weight 9.

1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

Filter 332. Total weight 81.

1	3	6	7	6	3	1
3	9	18	21	18	9	3
6	18	36	42	36	18	6
7	21	42	49	42	21	7
6	18	36	42	36	18	6
3	9	18	21	18	9	3
1	3	6	7	6	3	1

Filter 333. Total weight 729.

1	4	10	16	19	16	10	4	1
4	16	40	64	76	64	40	16	4
10	40	100	160	190	160	100	40	10
16	64	160	256	304	256	160	64	16
19	76	190	304	361	304	190	76	19
16	64	160	256	304	256	160	64	16
10	40	100	160	190	160	100	40	10
4	16	40	64	76	64	40	16	4
1	4	10	16	19	16	10	4	1

Filter 334. Total weight 6561.

1	5	15	30	45	51	45	30	15	5	1
5	25	75	150	225	255	225	150	75	25	5
15	75	225	450	675	765	675	450	225	75	15
30	150	450	900	1350	1530	1350	900	450	150	30
45	225	675	1350	2025	2295	2025	1350	675	225	45
51	255	765	1530	2295	2601	2295	1530	765	255	51
45	225	675	1350	2025	2295	2025	1350	675	225	45
30	150	450	900	1350	1530	1350	900	450	150	30
15	75	225	450	675	765	675	450	225	75	15
5	25	75	150	225	255	225	150	75	25	5
1	5	15	30	45	51	45	30	15	5	1

Filter 335. Total weight 59049.

Figure 4.1.1. Weighting of pixels achieved by repeated applications of a 3 x 3 window.

1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1

Filter 551. Total weight 25.

1	2	3	4	5	4	3	2	1
2	4	6	8	10	8	6	4	2
3	6	9	12	15	12	9	6	3
4	8	12	16	20	16	12	8	4
5	10	15	20	25	20	15	10	5
4	8	12	16	20	16	12	8	4
3	6	9	12	15	12	9	6	3
2	4	6	8	10	8	6	4	2
1	2	3	4	5	4	3	2	1

Filter 552. Total weight 625.

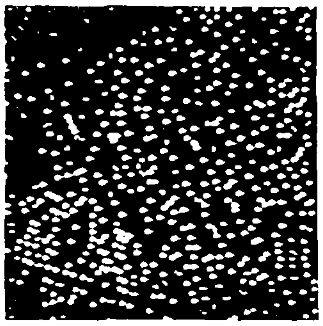
1	3	6	10	15	18	19	18	15	10	6	3	1
3	9	18	30	45	54	57	54	45	30	18	9	3
6	18	36	60	90	108	114	108	90	60	36	18	6
10	30	60	100	150	180	190	180	150	100	60	30	10
15	45	90	150	225	270	285	270	225	150	90	45	15
18	54	108	180	270	324	342	324	270	180	108	54	18
19	57	114	190	285	342	361	342	285	190	114	57	19
18	54	108	180	270	324	342	324	270	180	108	54	18
15	45	90	150	225	270	285	270	225	150	90	45	15
10	30	60	100	150	180	190	180	150	100	60	30	10
6	18	36	60	90	108	114	108	90	60	36	18	6
3	9	18	30	45	54	57	54	45	30	18	9	3
1	3	6	10	15	18	19	18	15	10	6	3	1

Filter 553. Total weight 15625.

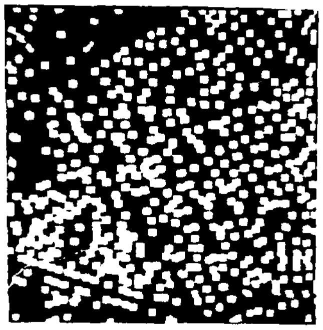
1	4	10	20	35	52	68	80	85	80	68	52	35	20	10	4	1
4	16	40	80	140	208	272	320	340	320	272	208	140	80	40	16	4
10	40	100	200	350	520	680	800	850	800	680	520	350	200	100	40	10
20	80	200	400	700	1040	1360	1600	1700	1600	1360	1040	700	400	200	80	20
35	140	350	700	1225	1820	2380	2800	2975	2800	2380	1820	1225	700	350	140	35
52	208	520	1040	1820	2704	3536	4160	4420	4160	3536	2704	1820	1040	520	208	52
68	272	680	1360	2380	3536	4624	11338	11678	11338	4624	3536	2380	1360	680	272	68
80	320	800	1600	2800	4160	11338	12298	12698	12298	11338	4160	2800	1600	800	320	80
85	340	850	1700	2975	4420	11678	12698	13123	12698	11678	4420	2975	1700	850	340	85
80	320	800	1600	2800	4160	11338	12298	12698	12298	11338	4160	2800	1600	800	320	80
68	272	680	1360	2380	3536	4624	11338	11678	11338	4624	3536	2380	1360	680	272	68
52	208	520	1040	1820	2704	3536	4160	4420	4160	3536	2704	1820	1040	520	208	52
35	140	350	700	1225	1820	2380	2800	2975	2800	2380	1820	1225	700	350	140	35
20	80	200	400	700	1040	1360	1600	1700	1600	1360	1040	700	400	200	80	20
10	40	100	200	350	520	680	800	850	800	680	520	350	200	100	40	10
4	16	40	80	140	208	272	320	340	320	272	208	140	80	40	16	4
1	4	10	20	35	52	68	80	85	80	68	52	35	20	10	4	1

Filter 554. Total weight 390625.

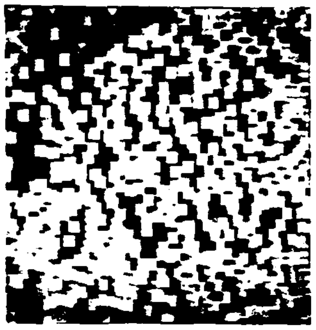
Figure 4.1.2. Weighting of pixels achieved by repeated applications of a 5 x 5 window.



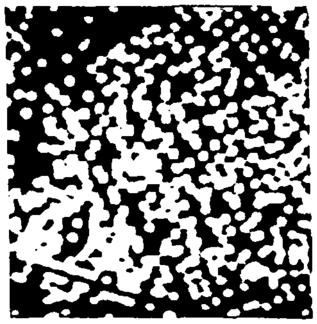
(a) Maxima pixels weighted.
Each region was given a weight of 255.
The different colors correspond to maxima
of different sizes. 536 regions.



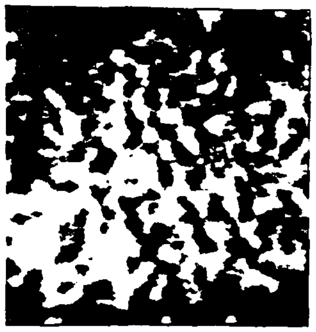
(b) One 3x3 window filter



(g) One 5x5 window filter

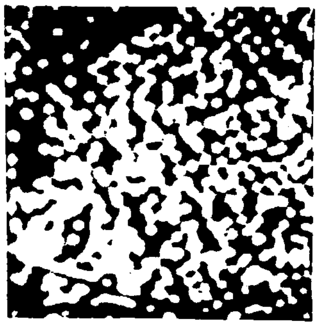


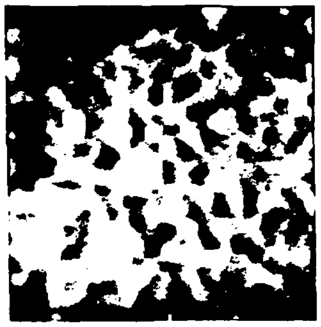
(c) Two 3x3 window filters

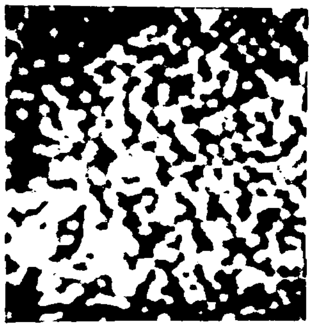


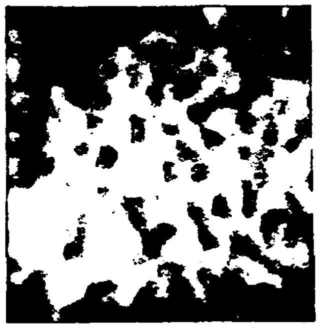
(h) Two 5x5 window filters

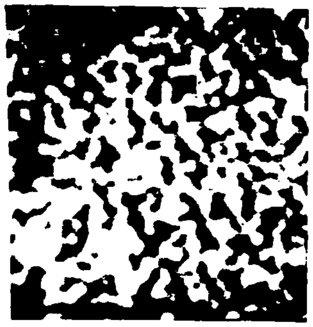
Figure 4.1.3. Density images from Maxima pixels.


(d) Three 3x3 window filters


(i) Three 5x5 window filters


(e) Four 3x3 window filters


(j) Four 5x5 window filters


(f) Five 3x3 window filters

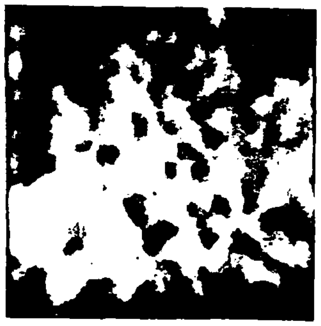

(k) Five 5x5 window filters

Figure 4.1.3 (continued)

measures. The fine texture of area at bottom up corresponds to the trailer court. The discontinuity in the rows of the trailer court is maintained if not emphasized in the density images.

Figure 4.1.4 shows the five 5 x 5 window filtered image level sliced. This shows the areas of different texture densities a little better.

On closer examination in this and pictures of Figure 4.1.3, a blocking or contouring effect may be discerned. This is owing to the integer truncation in the box filtering.

Figure 4.1.5 shows the weighted minima regions and the density images after five 3 x 3 and 5 x 5 iterations.

Figures 4.1.6 and 4.1.7 show the corresponding results of the above procedure as applied to the descending components of the maxima and the ascending components of the minima. The resulting density images are smoother than those of the corresponding extrema as the weight was distributed over more cells before the filtering began.

In Chapter 2.0 we had mentioned a texture transform procedure based on spatial co-occurrence matrices. The result of applying that to a quantized version of M4A is shown in Figure 4.1.8. This was done for comparison with the density images which can also be considered as texture transforms. The image in Figure 4.1.8 was generated assuming f in Definition 2.1.3 to be the identity function. The results do not match up over the entire image, indicating the two methods emphasize different texture attributes.

Antoher experiment was run to compare these results with the idea of using edges/unit area as texture measures. The edge image was obtained by applying the quick Roberts gradient function to the M4A-photo image. For a pixel (i,j) the quick Roberts gradient value is given by R(i,j):

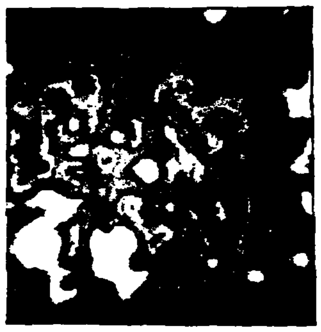
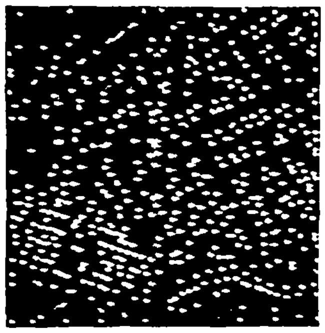
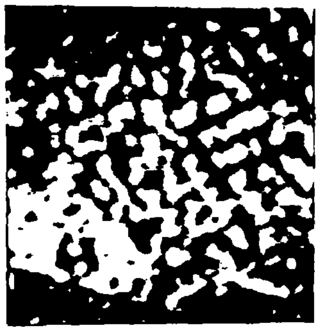


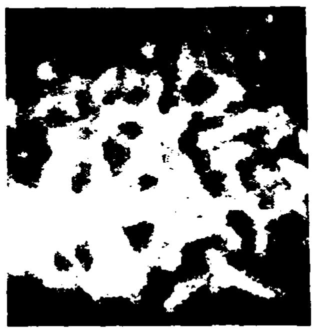
Figure 4.1.4. Weighted Maxima density image after applying a 5x5 averaging window five times and level slicing.



(a) Minima pixels weighted.
Each region was given a weight of 255.
The different colors correspond to minima
of different sizes. 511 regions.

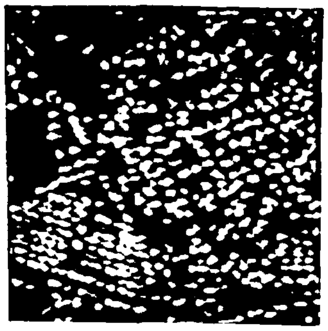


(b) Five 3x3 window filters

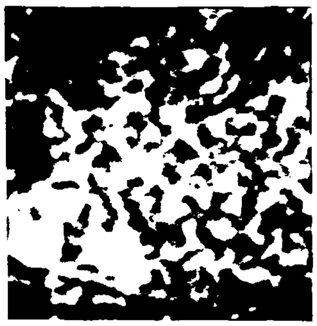


(c) Five 5x5 window filters

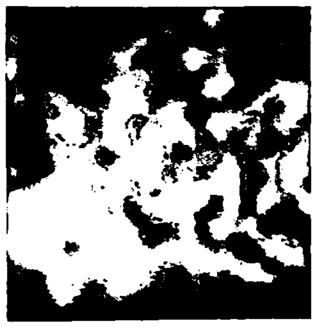
Figure 4.1.5. Density images from Minima pixels.



(a) Descending Component pixels weighted. Each component was given a weight of 255. The different colors correspond to components of different sizes. 536 regions.

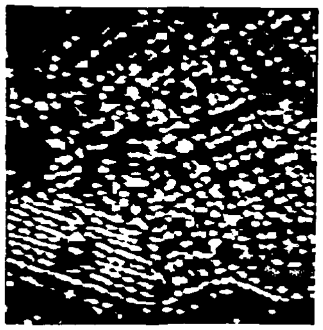


(b) Five 3x3 window filters

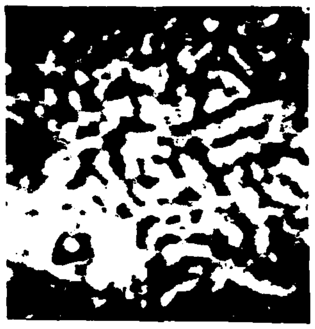


(c) Five 5x5 window filters

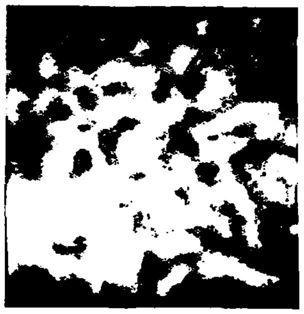
Figure 4.1.6. Density images from Descending Components.



(a) Ascending Components weighted.
Each component was given a weight of 255.
The different colors correspond to components
of different sizes. 511 regions.



(b) Five 3x3 window filters



(c) Five 5x5 window filters

Figure 4.1.7. Density images from Ascending Components.

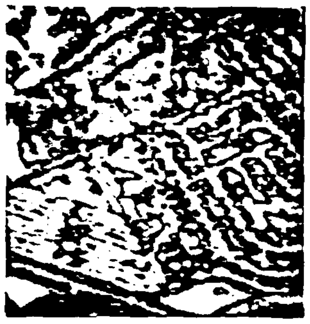


Figure 4.1.8 A Textural transform of M4A based on spatial co-occurrence matrices.

$$R(i,j) = |I(i,j) - I(i + 1, j + 1)| + |I(i + 1, j) - I(i, j + 1)|$$

This image is shown in Figure 4.1.9. To get the edge density image five 3 x 3 and 5 x 5 box filter operations were applied. These results are given in Figure 4.1.10. They compare quite well with the min/max density images.

Finally, texture density images were generated using both the minima and maxima. The full extrema image was obtained by relabeling the 2(max) label from the result of MNMX8 to the 1(min) label. Thus, all extrema were then marked by 1. A LBLCT operation was performed to give each maximally connected region a unique label to allow for the distributed weight procedure to be applicable. A point that was noted in this process was that the number of extrema obtained this way was not the sum of the relative minima and maxima. There were 511 minima and 536 maxima regions in M4A. The resulting extrema image above had only 976 regions. The difference is because in the process of labeling twos to ones, min and max extremum which happened to be adjacent (i.e., in each other's eight neighborhood) were merged into one region.

The weighted extrema image and corresponding density images are shown in Figure 4.1.11. These are similar to the min/max density functions as may be expected.

On a visual level the results seemed to capture texture pretty well. In the next section we will look at another method used to analyze the results.

4.2 Autocorrelation Tests

It was hoped that the histograms of the extrema density images would show three or four well-defined peaks corresponding to the three types of

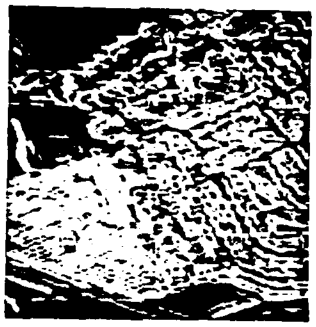
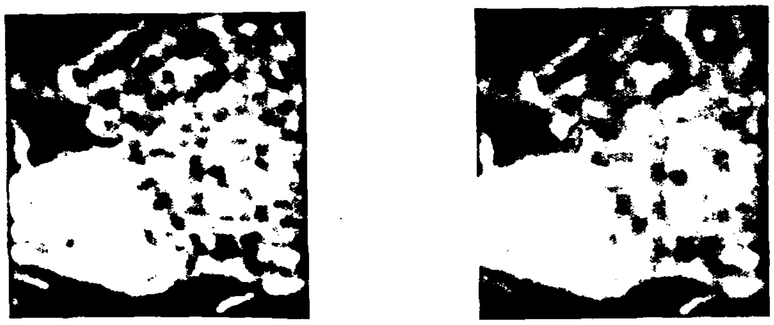
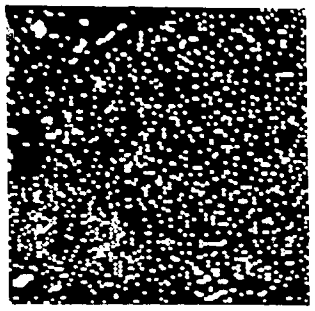


Figure 4.1.9 Roberts Gradient image of M4A.

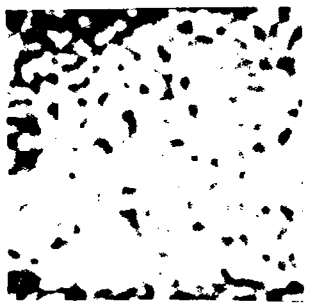


(a) Five 3x3 window filters (b) Five 5x5 window filters

Figure 4.1.10. Edge density images after applying averaging filters to the gradient image.



(a) Extrema (minima & maxima) pixels weighted.
Each region was given a weight of 255.
The different colors correspond to regions
different sizes. 976 regions.



(b) Five 3x3 window filters

(c) Five 5x5 window filters

Figure 4.1.11. Density images from Extrema pixels.

areas observed on the image. If this was so, then segmentation could be achieved based on level slicing on the histograms. In general, histograms of images with a small amount of filtering showed a number of sharply defined peaks. Histogram of images with high number of iterations had one large peak and some shallow peaks. None of these promised good direct segmentation. However, the histogram of the image which had been filtered three times with a 3 $\times$ 3 filter did have four peaks and seemed the best candidate for examination. Figure 4.2.1 shows this histogram.

In order to investigate the properties of the density measures, an experiment was performed in which the density images were quantized down to 2, 3, 4, and 5 levels. The resulting images were segmentations of the original based on extrema density. To examine how good the segmentations were, statistical properties of areas on the original image, defined by the segmented images, were obtained. The statistical properties were essentially autocorrelation values over the different areas. This function is discussed next.

.The autocorrelations computed did not contain cross product terms involving pixels from two different contiguous regions of a category. The spatial correlation for one category, and lag(L) is the ordinary correlation coefficient for two sets of measurements on a subset S of pixels in the category. This is described below.

Consider a maximal sequence of pixels in a row of the segmented image which belong to the same category. The first and last pixels in the sequence are considered to be on the boundary. If the sequence is N long, then for a given lag L, the right-most (N - L) pixels are in the subset S. S consists of all such pixels in all the connected sequences of the category. Note that the size of S decreases as L increases. The

```
1----------------------------------------------------------------------1
1                                                                      1
1                            HISTOGRAM OF                              1
1                                                                      1
1          M4AEXT CMB BND 4  03/29/79.  EXTREMA FILTER 333             1
1                                                                      1
1                     SUM OF FREQUENCIES =    16384                    1
1              LOW DATA VALUE =     0   HIGH DATA VALUE =    255       1
1                     MAXIMUM PROBAILITY = 0. 039                      1
1                MEAN =  0. 214E+02    VARIANCE =  0. 926E+03          1
1                                                                      1
1 PROBABILITY                                                          1
1  0. 032 I  ***                                                       1
1         I  ***                                                       1
1         I******                                                      1
1         I******                                                      1
1         I******                                                      1
1         I******                                                      1
1         I******                                                      1
1         I******                                                      1
1  0. 015 I*******                                                     1
1         I*******                                                     1
1         I*******                                                     1
1         I*******                                                     1
1         I*******                                                     1
1         I*******                                                     1
1         I*******                                                     1
1         I*******                                                     1
1  0. 009 I********                                                    1
1         I********                                                    1
1         I********                                                    1
1         I********                                                    1
1         I********                                                    1
1         I********                                                    1
1         I********                                                    1
1         I********                                                    1
1  0. 005 I********                                                    1
1         I********                                                    1
1         I*********                                                   1
1         I*********                                                   1
1         I*********                                                   1
1         I*********                     *                             1
1         I*********                     *                             1
1         I**********                    *                             1
1  0. 002 I**********                    *                             1
1         I**********                    *                             1
1         I********** *                  *                             1
1         I************** *              *              *      *       1
1         I************** *      **      *              *      *       1
1         I************** *  **  **      *      *       *   *  **      1
1         I************** *********  *   *      *       *********      1
1         I************** *********  *   *      *       *********      1
1         IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII 1
1          I     I     I     I     I     I     I     I     I     I      1
1          3     29    54    80    106   131   157   182   208   234    1
1                                                                      1
1                          DATA VALUES                                 1
1                                                                      1
1----------------------------------------------------------------------1
```

Figure 4.2.1. Histogram of extrema density image after three 3 x 3 window averaging filters.

corresponding gray tone in the original image for the pixel in S gives us the first measurement for the correlation count. The second is the gray tone of the pixel L columns to the left. We define the spatial autocorrelation $\alpha(L)$ on connected sequences which are run lengths along rows.

Definition 4.2.1

Let there be K connected sequences with length greater than L in a category. Let the size of each sequence be N_i. Also, let r_i be the image row for the i-th sequence and c_i be the column position in the image for the first pixel in sequence i. The spatial autocorrelation function for lag L is given by:

$$\alpha(L) = \left(\frac{1}{\#S} \sum_{i=1}^{K} \sum_{j=c_i+L}^{c_i+N_i-1} I(r_i,j)*I(r_i,\ j-L) - \mu_1\mu_2 \right) / \sigma_1\sigma_2,$$

where

$I(r_i,j)$ is the gray tone of the original image at pixel (r_i,j);

$$\mu_1 = \frac{1}{\#S} \sum_{i=1}^{K} \sum_{j=c_i+L}^{c_i+N_i-1} I(r_i,j) \text{ and}$$

$$\sigma_1^2 = \frac{1}{\#S} \sum_{i=1}^{K} \sum_{j=c_i+L}^{c_i+N_i-1} I(r_i,j)^2 \qquad - \mu_1^2$$

are the mean and standard deviation for the first set. Similarly:

$$\mu_2 = \frac{1}{\#S} \sum_{i=1}^{K} \sum_{j=c_i+L}^{c_i+N_i-1} I(r_i,\ j-L) \text{ and}$$

$$\sigma_2^2 = \frac{1}{\#S} \sum_{i=1}^{K} \sum_{j=c_i+L}^{c_i+N_i-1} I(r_i, j-L)^2 \qquad - \mu_2^2$$

are the mean and standard deviation for the second set in the correlation computation. The number of pixels in the set (#S) is given by:

$$\#S = \sum_{i=1}^{K} \sum_{j=c_i+L}^{c_i+N_i-1} 1$$

The segmentation of the density images were created by three rules. First, by an equal interval quantization. Since the density tended to be concentrated at the low end of its range, the segments of low density tended to cover most of the area. Each of the 3 x 3 filter iterations on the extrema image were semgented.

Second, the equal probability quantization rule was tried. This produced images with approximately equal number of pixels in each segment. As above, each of the 3 x 3 filter iterations extrema density images were segmented to 2, 3, 4, and 5 levels.

By the third rule the image which had to be filtered three times with a 3 x 3 window was clustered into 4 segments corresponding to the 4 peaks in its histogram. All these rules generated a large number of tables and values. We will examine the results of the last rule in more detail. The segmented images are shown in Figure 4.2.2.

As mentioned above, the autocorrelation function computed the autocorrelations in the horizontal direction as well as the mean and standard deviation of the gray tones for the entire category. The mean corresponded to the average height within a segment. The standard deviations and autocorrelations measured aspects of texture within a segment. Table 4.2.1

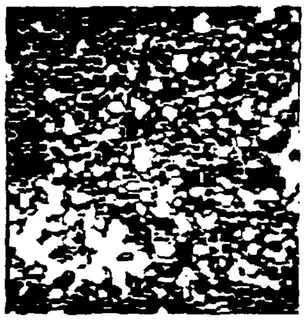


Figure 4.2.2a. Four segments (0-black), (1-green), (2-orange) and (3-purple) of the extrema density image after three 3x3 averaging filters.

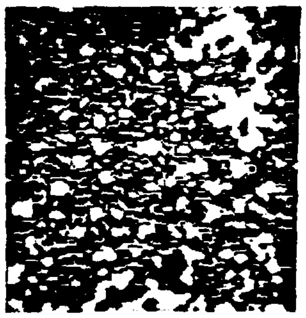


Figure 4.2.2b. Transpose of filter 333 extrema density image.

shows the autocorrelations for the four categories for different lag values; for both horizontal and vertical scans. The results for the vertical scans were obtained by applying the same algorithms to the images after they were transposed (i.e., rows and columns interchanged).

It was expected that high standard deviations as measured of height variability tend to indicate rough texture. Autocorrelations of pixel heights on images generally tend to be high for small lags and drop to near zero for larger lags. Rapidly decreasing autocorrelations with increasing lag indicates rough texture.

The results examined over the various segmentations were mixed. In general, the means and standard deviations increased slowly with increasing extrema density of a segment. This trend in standard deviations supports the idea that extrema density measures texture. The fact that means and standard deviations increase together and are therefore correlated, supports the observation that extrema images are like contour maps.

There was no clear pattern between segments and autocorrelations. The autocorrelations in the vertical directions were significantly different from those in the horizontal direction for the same segmentation. This was perhaps due to a directionality in the textures in M4A.

A statistical problem whose significance is not clear yet is the fact that the number of pixels in the subset S goes down as L increases. That is, the autocorrelations at different lags are being computed for different sample sizes. As long as the sample size is large, this would not have much effect. However, for some categories the sample size falls off quite rapidly with increasing lag. This may give rise to statistically invalid results and may explain why there is a slight upswing in the autocorrelation for higher lags.

```
AUTO-CORRELATIONS FOR SEGMENT NO.   0

          MEAN GRAY TONE = 36.40
          STD.  DEVIATION = 12.67
          SEGMENT SIZE    =  2149 PIXELS.

          I--------------------II--------------------I
          I     HORIZONTAL     II      VERTICAL      I
I------I---------I----------II---------I----------I
I  LAG I  ALPHA  I  SAMPLE  II  ALPHA  I  SAMPLE  I
I------I---------I----------II---------I----------I
I   1  I   0.93  I   1609   II   0.92  I   1577   I
I   2  I   0.80  I   1167   II   0.80  I   1101   I
I   3  I   0.65  I    875   II   0.70  I    774   I
I   4  I   0.44  I    682   II   0.62  I    538   I
I   5  I   0.23  I    544   II   0.55  I    385   I
I   6  I   0.03  I    439   II   0.52  I    274   I
I   7  I  -0.10  I    358   II   0.49  I    197   I
I   8  I  -0.13  I    289   II   0.54  I    138   I
I   9  I  -0.14  I    234   II   0.64  I     97   I
I  10  I  -0.11  I    183   II   0.73  I     68   I
I------I---------I----------II---------I----------I


AUTO-CORRELATIONS FOR SEGMENT NO.   1

          MEAN GRAY TONE = 39.03
          STD.  DEVIATION = 13.25
          SEGMENT SIZE    =  4094 PIXELS.

          I--------------------II--------------------I
          I     HORIZONTAL     II      VERTICAL      I
I------I---------I----------II---------I----------I
I  LAG I  ALPHA  I  SAMPLE  II  ALPHA  I  SAMPLE  I
I------I---------I----------II---------I----------I
I   1  I   0.90  I   2383   II   0.86  I   2447   I
I   2  I   0.73  I   1367   II   0.68  I   1443   I
I   3  I   0.61  I    795   II   0.58  I    876   I
I   4  I   0.51  I    485   II   0.48  I    550   I
I   5  I   0.43  I    301   II   0.43  I    344   I
I   6  I   0.38  I    190   II   0.39  I    211   I
I   7  I   0.35  I    118   II   0.33  I    127   I
I   8  I   0.34  I     69   II   0.28  I     74   I
I   9  I   0.32  I     37   II   0.25  I     43   I
I  10  I   0.46  I     16   II   0.22  I     24   I
I------I---------I----------II---------I----------I
```

Table 4.2.1. Autocorrelations ALPHA as a function of LAG for the four segments of Figure 4.2.2. SAMPLE indicates the number of pixels for computing the autocorrelation for different lags.

AUTO-CORRELATIONS FOR SEGMENT NO. 2

MEAN GRAY TONE = 41.80
STD. DEVIATION = 15.36
SEGMENT SIZE = 6957 PIXELS.

	HORIZONTAL		VERTICAL	
LAG	ALPHA	SAMPLE	ALPHA	SAMPLE
1	0.90	4980	0.84	5118
2	0.71	3443	0.59	3631
3	0.44	2372	0.40	2581
4	0.19	1668	0.29	1875
5	0.04	1183	0.24	1372
6	-0.05	833	0.22	1000
7	-0.04	584	0.21	732
8	-0.01	430	0.20	541
9	0.01	310	0.15	407
10	0.02	232	0.14	304

AUTO-CORRELATIONS FOR SEGMENT NO. 3

MEAN GRAY TONE = 45.16
STD. DEVIATION = 16.10
SEGMENT SIZE = 3184 PIXELS.

	HORIZONTAL		VERTICAL	
LAG	ALPHA	SAMPLE	ALPHA	SAMPLE
1	0.90	2358	0.74	2444
2	0.73	1662	0.48	1813
3	0.58	1131	0.57	1324
4	0.47	774	0.58	959
5	0.35	536	0.42	723
6	0.27	385	0.42	552
7	0.25	284	0.50	422
8	0.31	217	0.32	321
9	0.41	169	0.22	248
10	0.55	131	0.39	188

Table 4.2.1 (continued). Autocorrelations ALPHA as a function of LAG for the four segments of Figure 4.2.2. SAMPLE indicates the number of pixels for computing the autocorrelation for different lags.

In view of the above discussion, various suggestions emerge. First, some similar experiments should be tried on larger images with clearly identifiable texture and no directionalities, or balanced directionalities. Artifically synthesized texture images could be useful. Second, a part of the reason for lack of clearly identifiable patterns with autocorrelations is that many areas may be sloped. In these areas, where gray tones locally approximate a tilted plane, pixels will be significantly positively or negatively correlated. It may be worthwhile to check this idea by using sloped facet filtering models or experiment with artificial images with tilted planes in them.

The weight distribution over extrema (see Section 4.1) are inversely proportional to extrema size and may not be best for creating extrema density images. Consider a large area which is perfectly flat and is a local extrema. This area has a smooth (or no) texture, but will be given a higher density measure than a smooth slope. Weights chosen inversely proportional to a higher power of extrema may yield an improved texture density image.

5.0 ATTRIBUTES OF PRIMITIVES

Texture measures based on histogram counts or density images tie together structural and statistical approaches. The approach is structural in the sense that the primitives are structural. The approach is statistical in that the spatial interaction, or the lack of it, between primitives is measured by probabilities. These are all weak texture measures and can be useful in segmentation when the texture patterns in the image have weak spatial interaction. For textures which have strong spatial interactions it may be necessary to determine, for each pair of primitives, the frequency with which the primitives co-occur in a specified spatial relationship. These would be the strong texture measures mentioned in Chapter 2. Meanwhile, a stronger form of the weak features can be explored by taking into account properties of primitives or the distribution of these properties. A good example is the work of Mitchell and Carlton (1978) who not only used the frequency of occurrence of the extrema primitive (density) but also one of its attributes, the height.

Different primitives, of course, yield different attributes. As a matter of fact, a primitive may be described as a connected set of resolution cells characterized by a list of attributes. In the rest of the chapter we will look at some of the attributes of the extrema and the reachability set primitives. Since our aim is segmentation of the image, we will look at a general scheme to extract the primitive attributes and an unsupervised clustering method to cluster the primitives. This will then allow us to define a segmentation on the image. In what follows we will use the term primitive to mean both the extrema as well as the reachability sets.

The last section discusses some extensions of attributes for strong texture measures.

5.1 Extrema Properties

We have already encountered one primitive attribute in the last chapter - the size of the primitive. We had used the size of the frequency of the primitives to create the weight distributed images for the generation of the density images. The size is an important attribute, for in addition to being one, it is also used in the definition of some other attributes.

The next most intuitive attribute is perhaps the height. This better described by the one-dimensional cross-section shown in Figure 5.1.1. This example is a maxima or a descending component of a maxima. "h_y", the relative height is the difference between the maxima and its highest valley. "h_a" is the absolute height of the maxima. "h_y" is related to contrast while "h_a" is related to brightness or intensity at that pixel value. The width or size associated with a maxima can be the distance between its two adjacent minima or valleys. Corresponding definitions exists for valley heights (depths) for the minima and the ascending component primitives.

Ehrich and Foith (1976, 1978) used maxima values (heights) in horizontal directions as primitives. They represented the primitives using the technique of a relational tree, which is an elegant method of storing attribute values of primitives, and at the same time showing the recursively nested structure of the primitives.

When we go to two-dimensional extrema, some modifications were made depending upon the property. We had wanted the property extraction to be a one or two pass operation on the image (this general scheme is described later). In order to determine widths and relative heights in two dimensions in an analogous manner as for the one-dimensional example above,

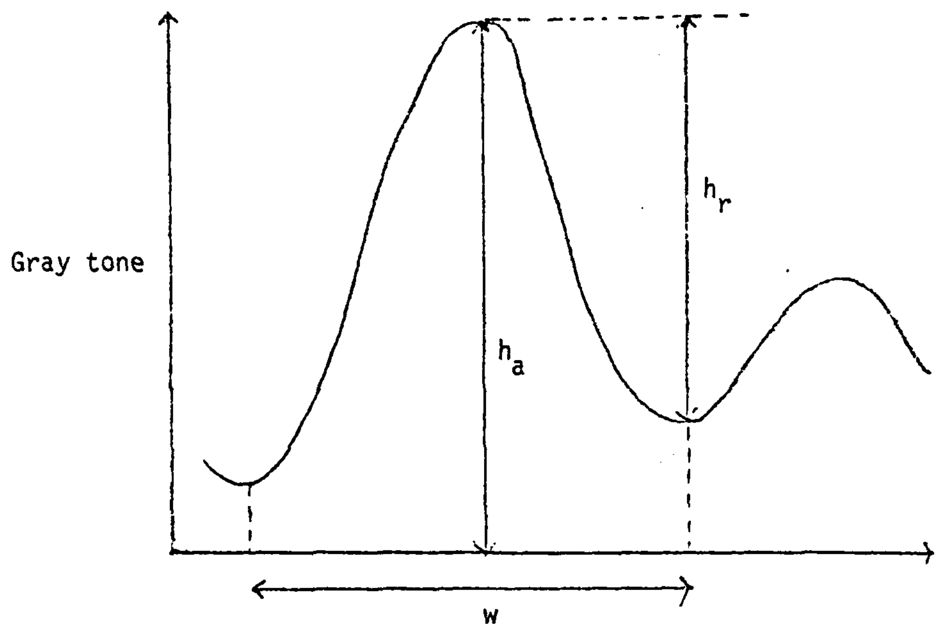


Figure 5.1.1. Showing absolute height "h_a", relative height "h_r" and width "w" of a maxima, in a one-dimensional cross-section.

several iterations would be needed, so the definitions of relative height and primitive width were modified. For the reachability sets the relative height become the difference between the maxima and the minimum gray tone. This is the number of gray tones in the region. Note that this does not guarantee the minimum gray tone to be one of the boundary pixels of the reachability set. The width was characterized by the shape (elongation or circularity) of the reachability set. The relative height and width for an extrema were not used. Another attribute used was the average gray tone for an extrema. This is again meaningful for the reachability sets only.

An attribute of the primitives which was useful indirectly was the center of mass of the regions. This is the pixel location of the center of mass of each region and is defined below.

Definition 5.1.1

The center of mass of a subset S of the image, denoted by (r_m, c_m) is given by:

$$r_m = \frac{1}{\#S} \sum r_i \quad \text{and} \quad c_m = \frac{1}{\#S} \sum c_i \text{ for all } (r_i, c_i) \in S$$

The center of mass for single pixel regions was the pixel itself. The center of mass came in useful in computing other properties. It was also designed for input to the region adjacency program discussed at the end.

Other attributes examined dealt with shape and orientation. The simplest definitions of the spread of a region are the dimensions and perhaps area of the covering rectangle. A more realistic definition is elongation. This can be defined as the ratio of the larger to smaller eigen value of the 2 x 2 second moment matrix obtained from the $\binom{x}{y}$

coordinates of the pixels in the regions (Bachi, 1973; Frolov, 1975). Actually the square root of the ratio corresponds to the ratio of the major to the minor axes of the covering ellipse. When the region is circular, this value is 1. In this computation, the $\binom{x}{y}$ coordinates are measured relative to the center of the region. Thus, having an image with the center of mass pixels for each region marked out makes this computation quite straight forward.

Another measure for circularity is the ratio of the standard deviation to the mean of the radii from the region center to its border (Haralick, 1975). However, this was not used as many of the regions had only a few pixels which made the measure difficult for most part.

As can be seen from the images of the reachability sets, they are not usually circular in shape but also have specific orientations. Two measures for this were examined. The first was based on the second moment matrix of the pixels described above. The components of the normalized eigen vectors corresponding to the largest eigen value are the direction sine and cosine for the region. These angles can be quantized over certain ranges, yielding the orientation of the shape.

The second orientation measure was different in more than one way from the attributes discussed above. It is actually a strong measure. In this case the orientation angle for each region was defined by the proximity and location of the closest region to it. Thus, this was a co-occurrence measure. The proximity was measured between the center of masses for the two regions. If the closest region was too far away, a special value was assigned for the orientation property.

There are many more complex properties that could be defined, but this would have made the investigation an unending task. In the next section a general scheme is presented to extract the properties. This is followed by a discussion on how the properties may be stored and used.

5.2 Property Extraction

Because of a large number of primitives per image and a variety of properties, the property extraction programs were restricted to computing properties that took one or two scans of the image. A small 128 x 128 image contained 500-800 primitives. Owing to the limited core on the minicomputer, a satisfactory scheme was developed which was based on two assumptions. The first was that there would be enough memory available to hold a few rows of the image and values for about 200 primitives at a time. The second and more important assumption was that the unique region labels assigned to the primitives were partially ordered. This implies that at any row of the image if the lowest label encountered is n, then from then on fill the end of the image, no labels with values less than n would be encountered. This is exactly the way the REACH algorithm assigns labels as it scans from top to bottom. Actually the restriction does not have to be followed strictly as long as the labels are ordered enough when a "shrinking" condition occurs. The program keeps track of property values in a buffer for regions as it scans along. If a region label is encountered which would bring the buffer close to overflowing, a "shrink" operation is called upon. This writes out to the disc the property values for regions that have been completed, thus shrinking or freeing up buffer space. The values written out are for regions with labels one less than the lowest label in the current row.

The scheme works extremely well and large images are processed without any problems. The properties discussed before can be extracted by this method. For example, size implies counting pixels of different regions as the scanning progresses. Height is computed by keeping track of the largest and least greytone of a region. Center of mass components are computed by summing the row and column values for the region. Many of the properties can be computed simultaneously in one scan. The second moment matrix computation requires reading in the center of mass image at the same time thus making it effectively a two-pass operation.

The property values are written out in tabular form to the disc. This is essentially the form required by the clustering algorithm. However, if a "property image" needs to be created, it may be done using the property values table and if necessary the images with the primitives. The center of mass image is one such example. The centers of mass region were read in from the table and an image created with the corresponding locations marked. Figure 5.2.1 shows the center of mass images for the reachability set primitives.

Property images may be created from any of the other properties as well. Take for example the height property. We can create an image in which we can mark in each pixel in a primitive region of size N the value h indicating that it is a part of a primitive having height h. [illegible]ternatively, we can mark it h/N, indicating its contribution to the [illegible]ve area or perhaps mark the center of mass pixel with the value [illegible] [illegible]ages may be viewed to see the distribution of the height

[illegible] to the property image concept is the property density [illegible]ilar process to the frequency density image of [illegible] shows an image for which a value W•h has

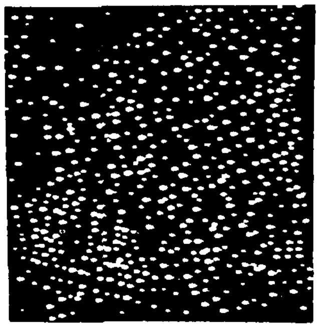


Figure 5.2.1a. Center of Mass pixels for the Descending Component regions.

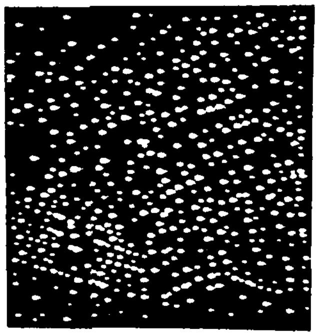


Figure 5.2.1b. Center of Mass pixels for the Ascending Component regions.

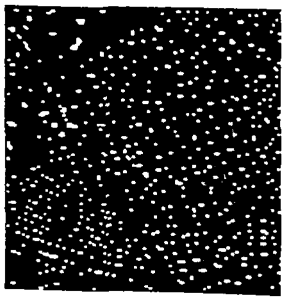


Figure 5.2.2a. Absolute Height property image for M4A. (Maxima Primitive)

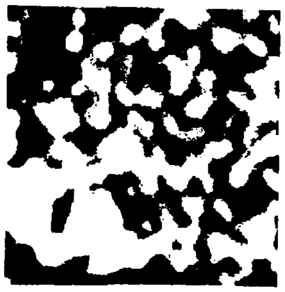


Figure 5.2.2b. Absolute Height property density image obtained by applying five 5x5 window averaging iterations.

assigned to each N cells of the maxima primitive. 'h' is the height and W a weight. This image was iteratively defocused and the resulting property density image is also shown. This like the other density images, can be used as a texture information band in multispectral clustering.

In order to examine and get a feel for the distribution of the property values, another general set of programs were written to compute and print out histograms, bar graphs, or the data in tabular form. This helped very much in the selection of properties for clustering. Figure 5.2.3 shows examples of these.

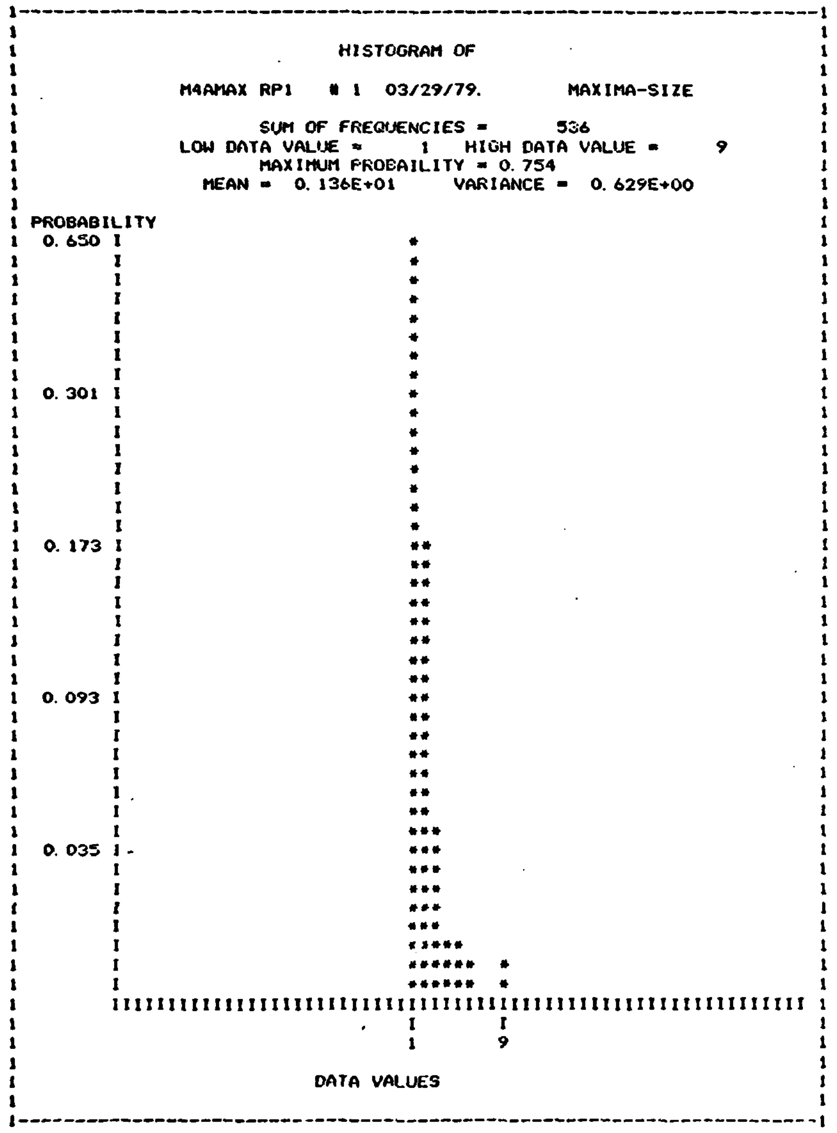
Owing to lack of time, not much experimenting was done with property density images. Instead, direct clustering was applied to the primitives themselves. We discuss this in the next section.

5.3 Clustering

Clustering calls for the grouping of similar primitives into clusters based on their attribute values. The clustering function is not an image operator in that its domain is the attribute value table. It assumes that any spatial information if necessary has already been included in the attribute value table. Only when the final clusters have been determined, is the primitive image accessed for relabeling.

Owing to the large number of cases (primitives) to be clustered and lack of ground truth, an unsupervised clustering scheme was devised which was stringent on core memory. In what follows we will discuss the clustering method and the definition and properties of these clusters.

Suppose we have S primitives and each of these primitives is characterized by K attribute values. The K attribute values are referred to as the



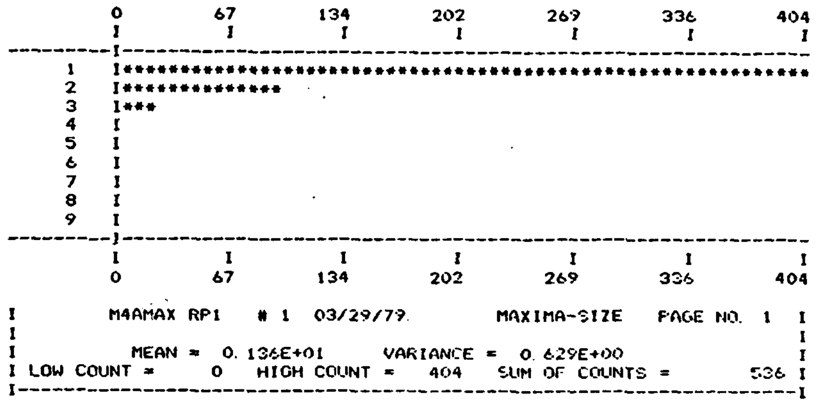
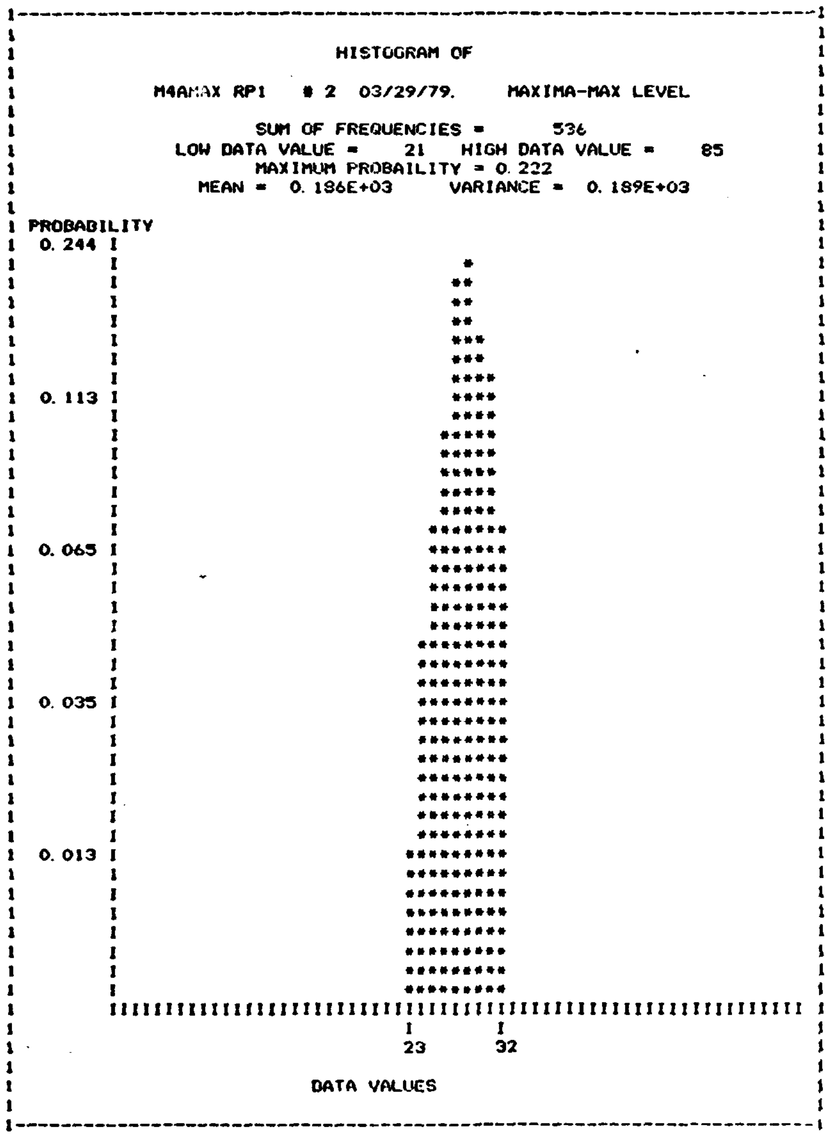


Figure 5.2.3a. Histogram and bargraph of sizes for maxima of M4A. Size is in number of pixels. The plots show how many times maxima of different sizes occurred in M4A. The histogram shows this in terms of probability while the bargraph shows the actual counts.



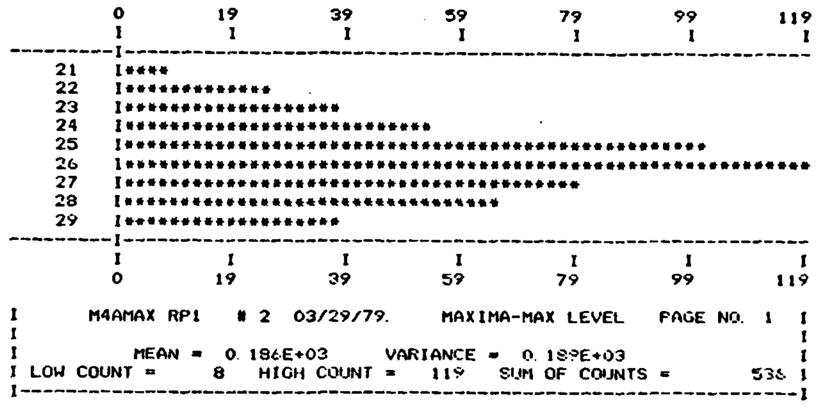
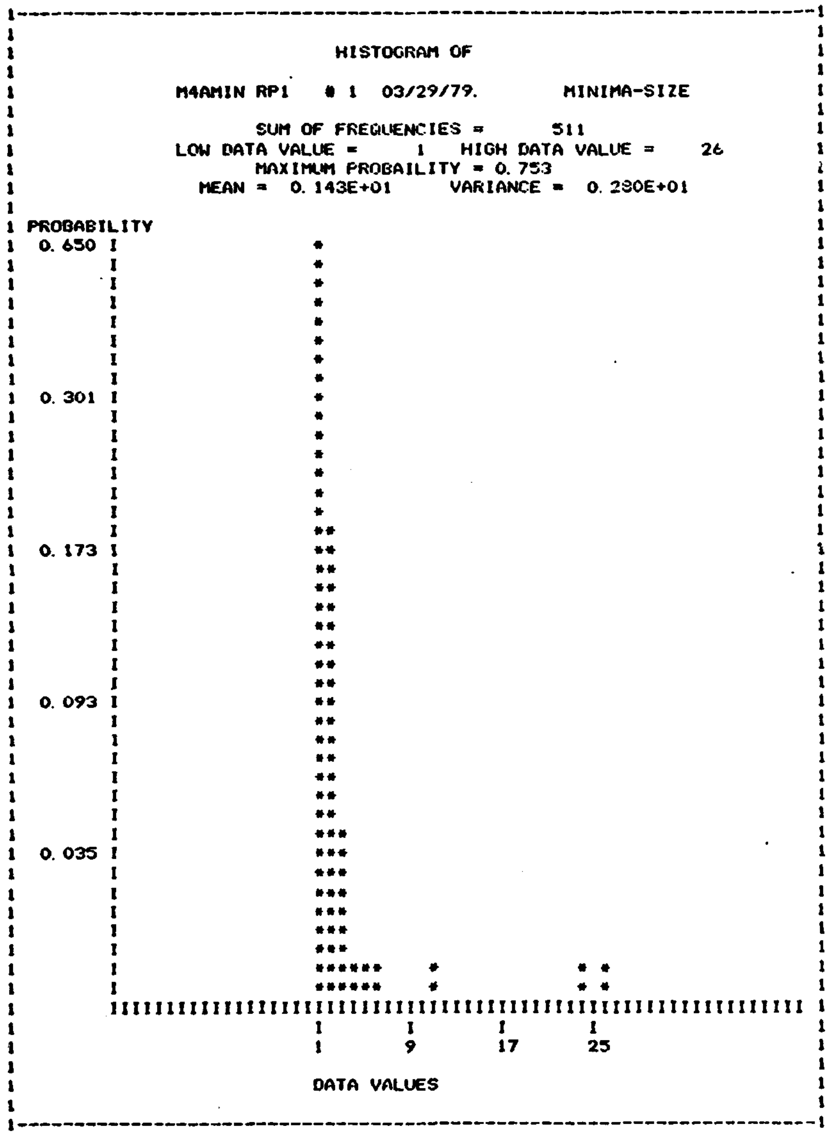


Figure 5.2.3b. Histogram and bargraph of grey levels (absolute heights) for the maxima of M4A. Plots show the distribution of the absolute height attribute.



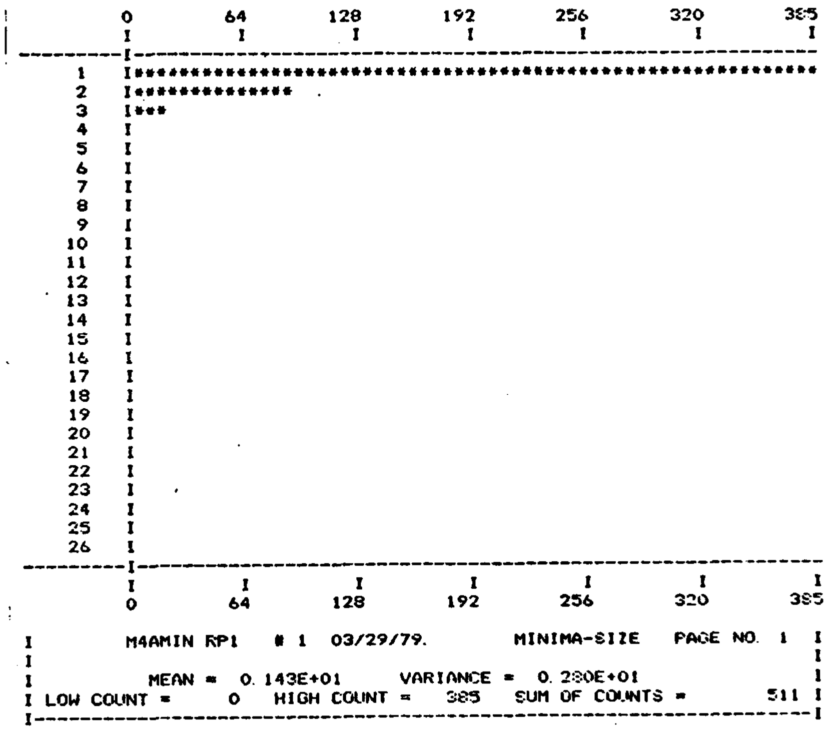
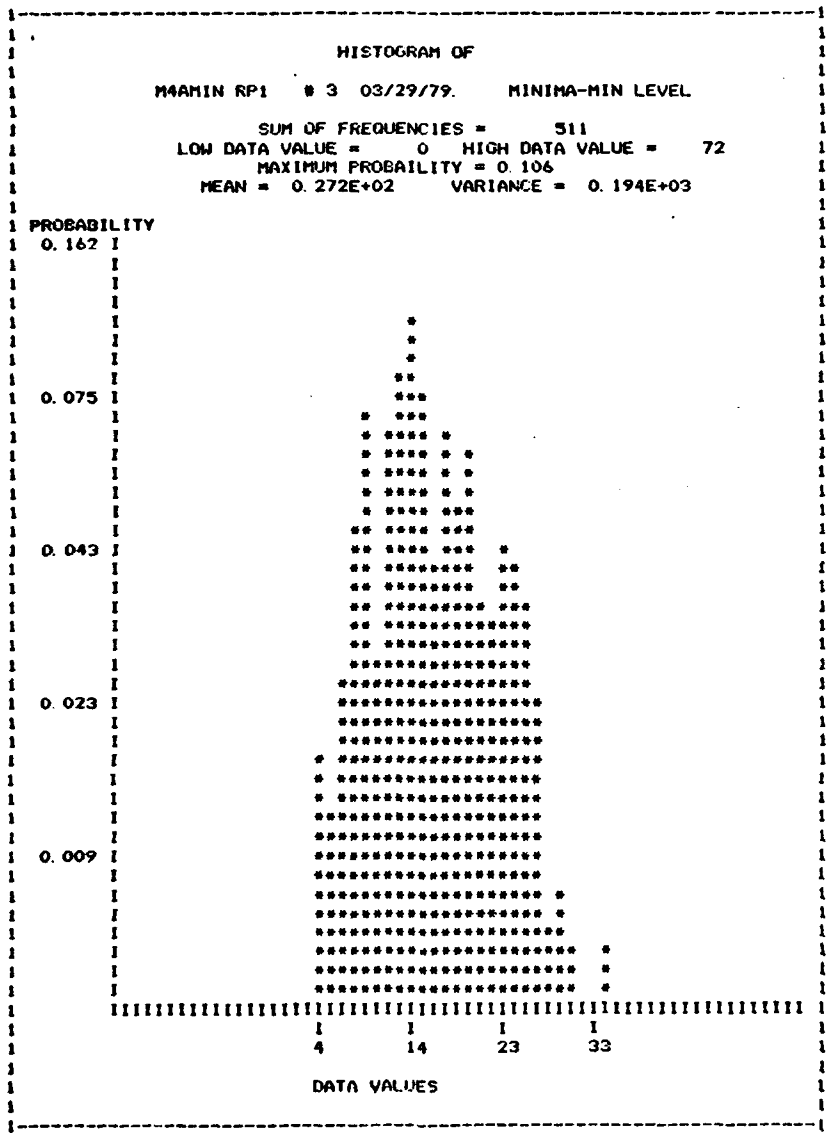


Figure 5.2.3c. Histogram and bargraph of sizes for minima of M4A. Size is in number of pixels.



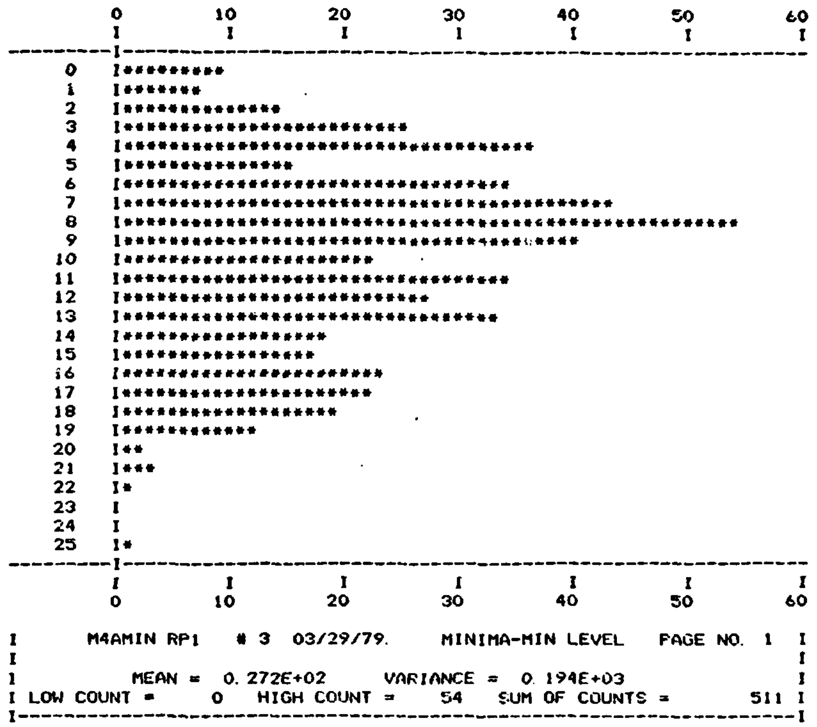
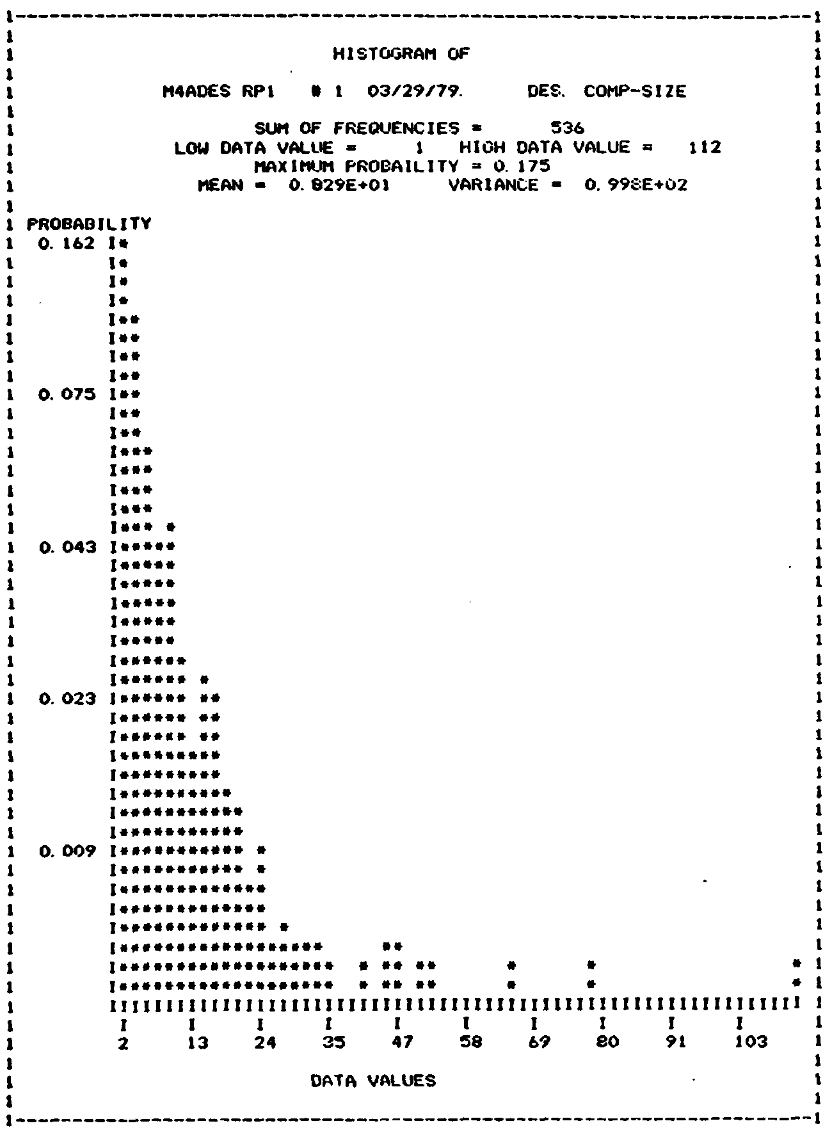


Figure 5.2.3d. Histogram and bargraph of grey levels (absolute heights or depths) for minima of M4A.



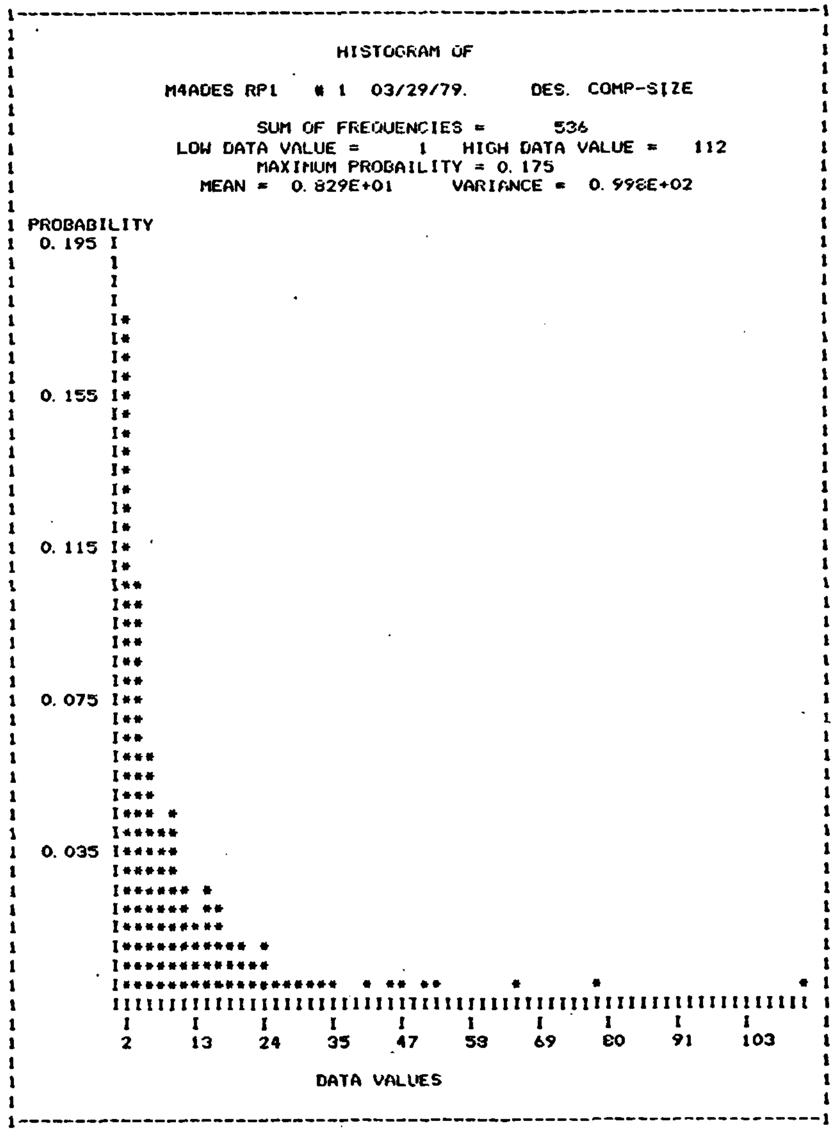
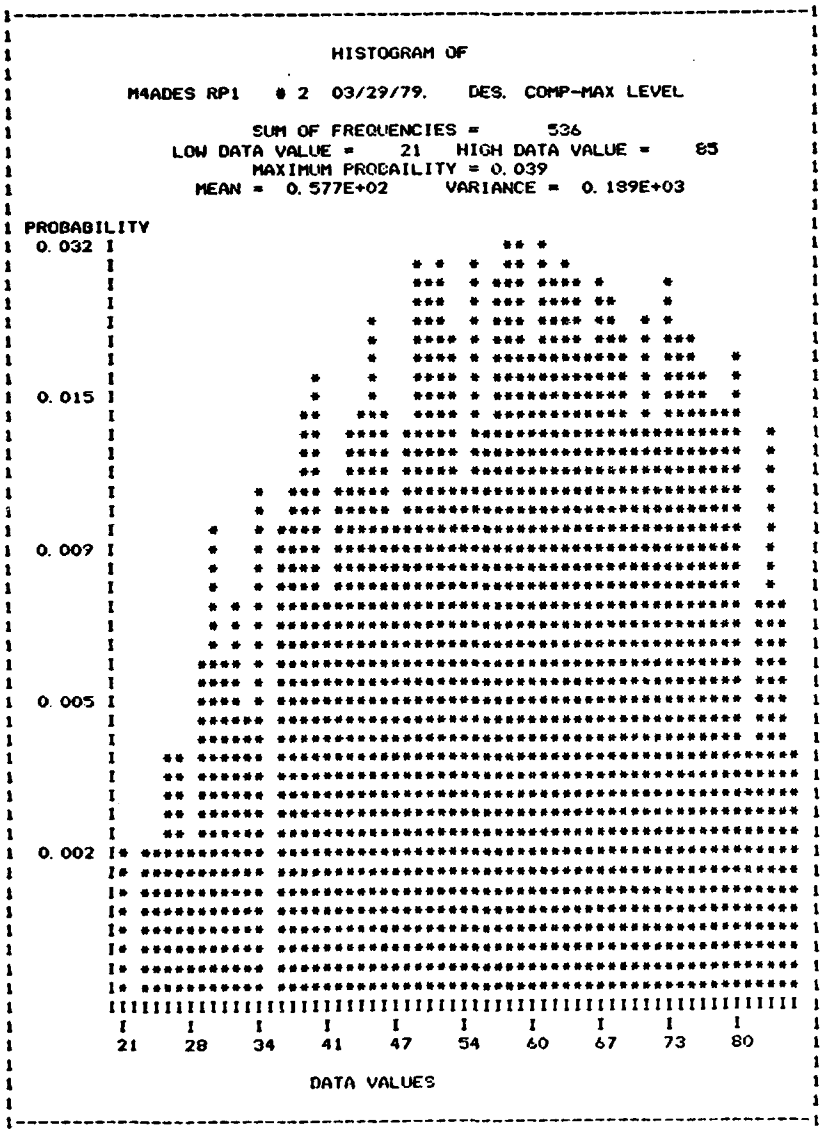


Figure 5.2.3e. Histograms of sizes of the descending components of the maxima of M4A. Shown both in log and regular scale. Size is in number of pixels. The log scale allows for the examination of the wide range of the distribution.



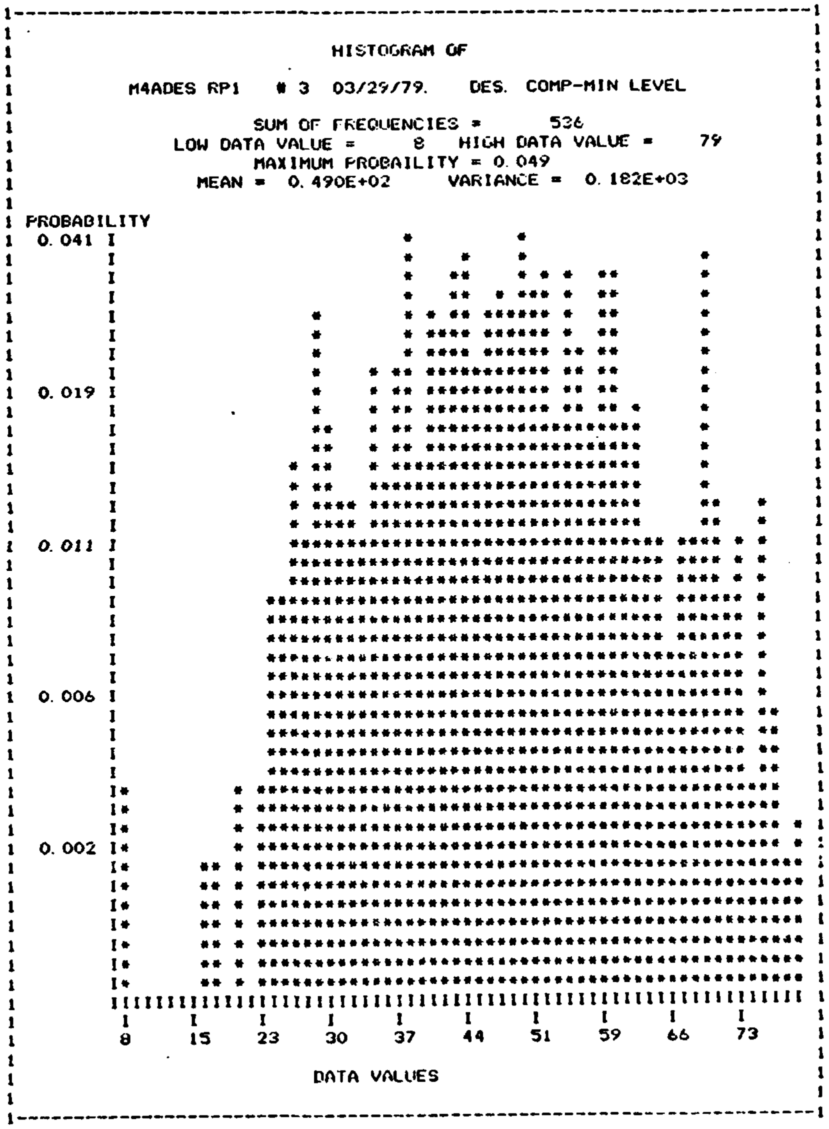
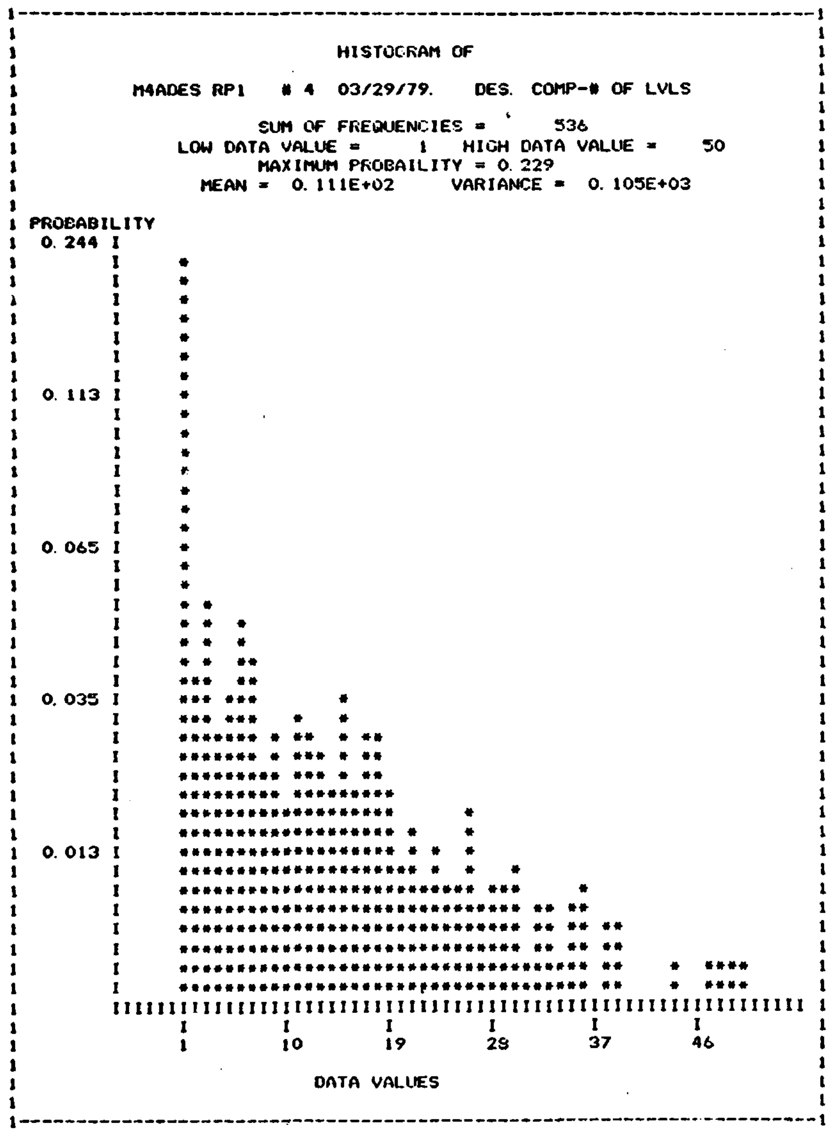


Figure 5.2.3f. Histograms of maximum and minimum grey levels for the descending components of the maxima of M4A.



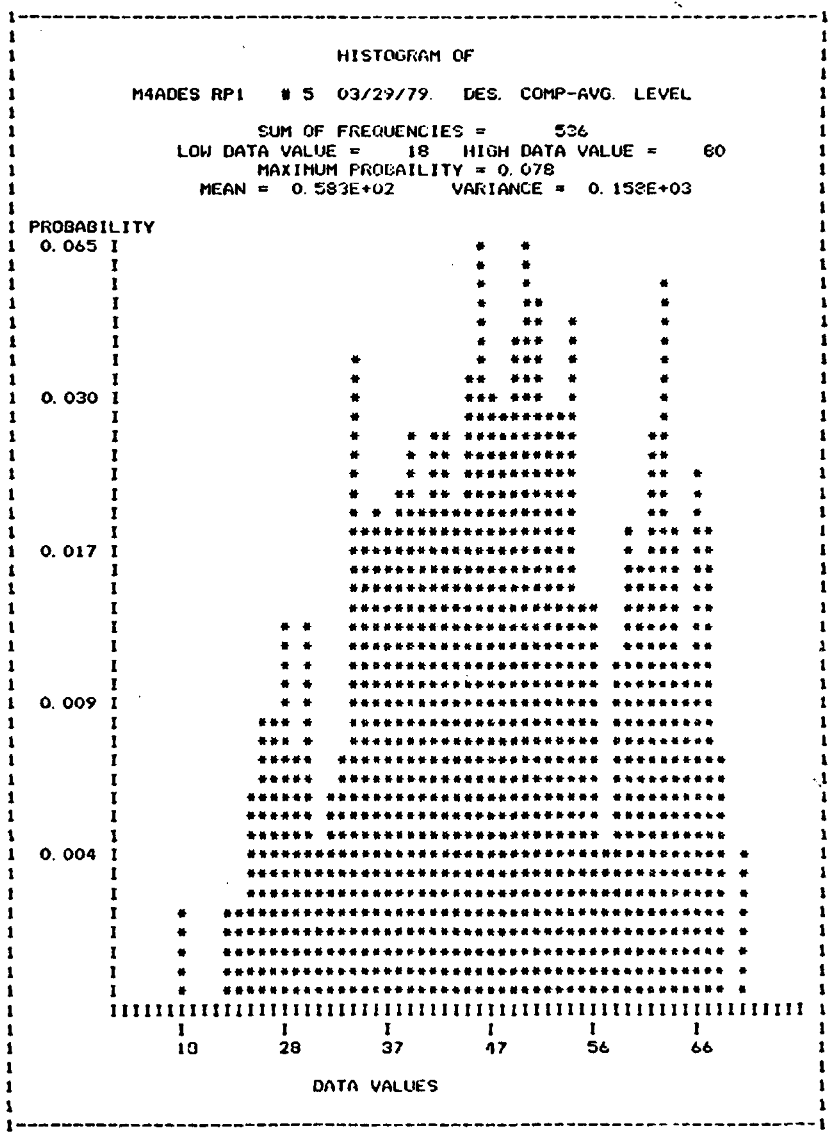
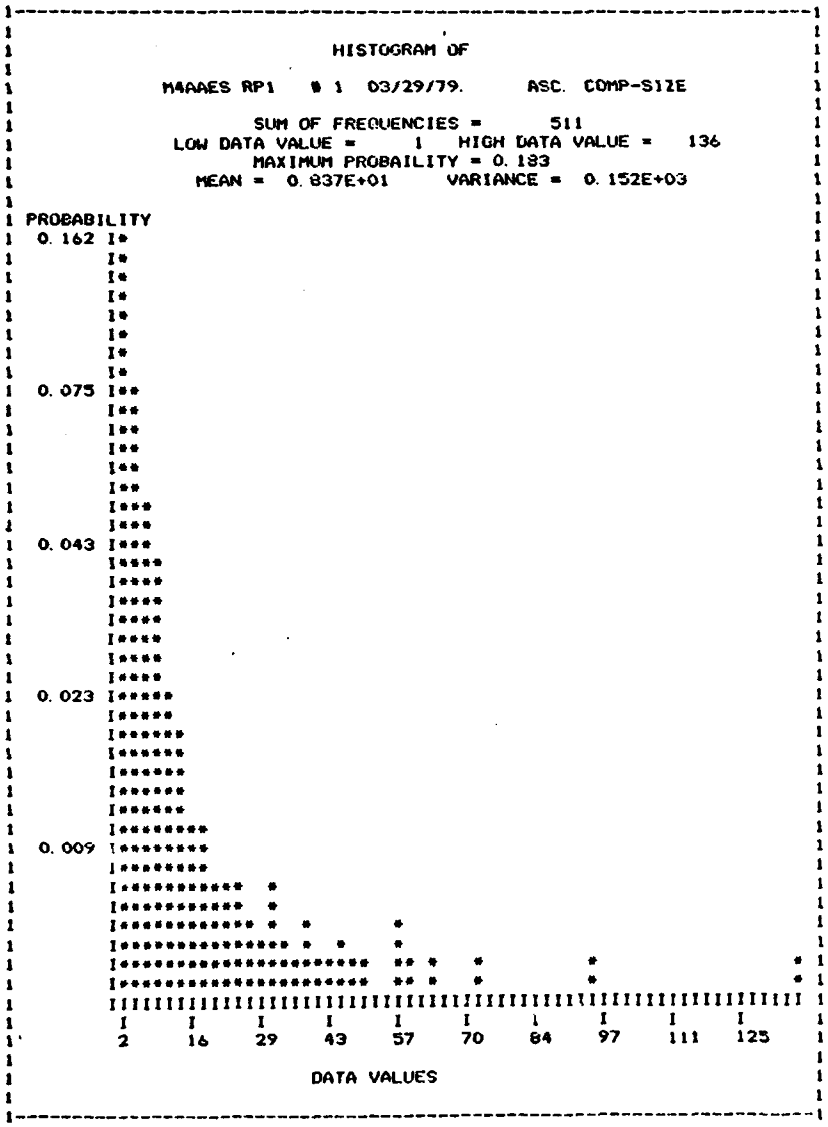


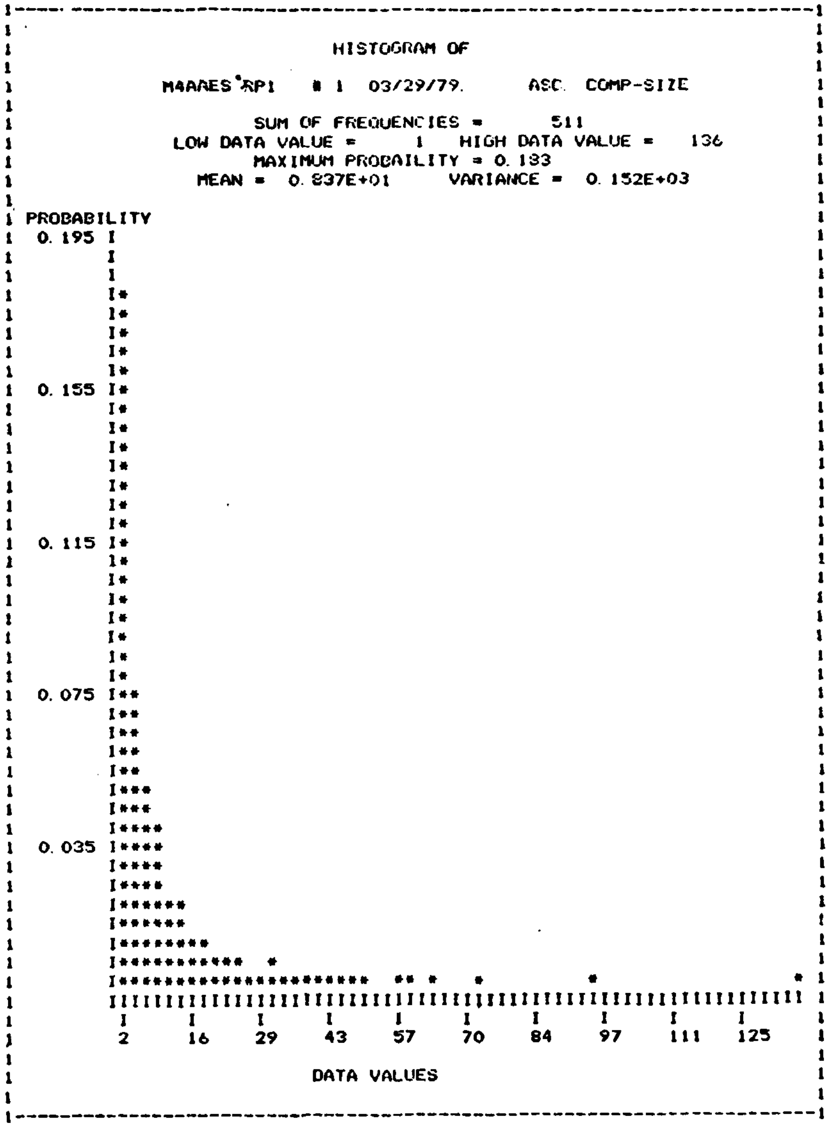
Figure 5.2.3g. Histograms of the number of levels (relative height) and the average grey level (average absolute height) for the descending components of the maxima of M4A.



```
                 HISTOGRAM OF

M4AAES RP1   # 1  03/29/79.      ASC. COMP-SIZE

           SUM OF FREQUENCIES =       511
     LOW DATA VALUE =      1   HIGH DATA VALUE =   136
           MAXIMUM PROBAILITY = 0.183
        MEAN =  0.837E+01      VARIANCE =  0.152E+03
```

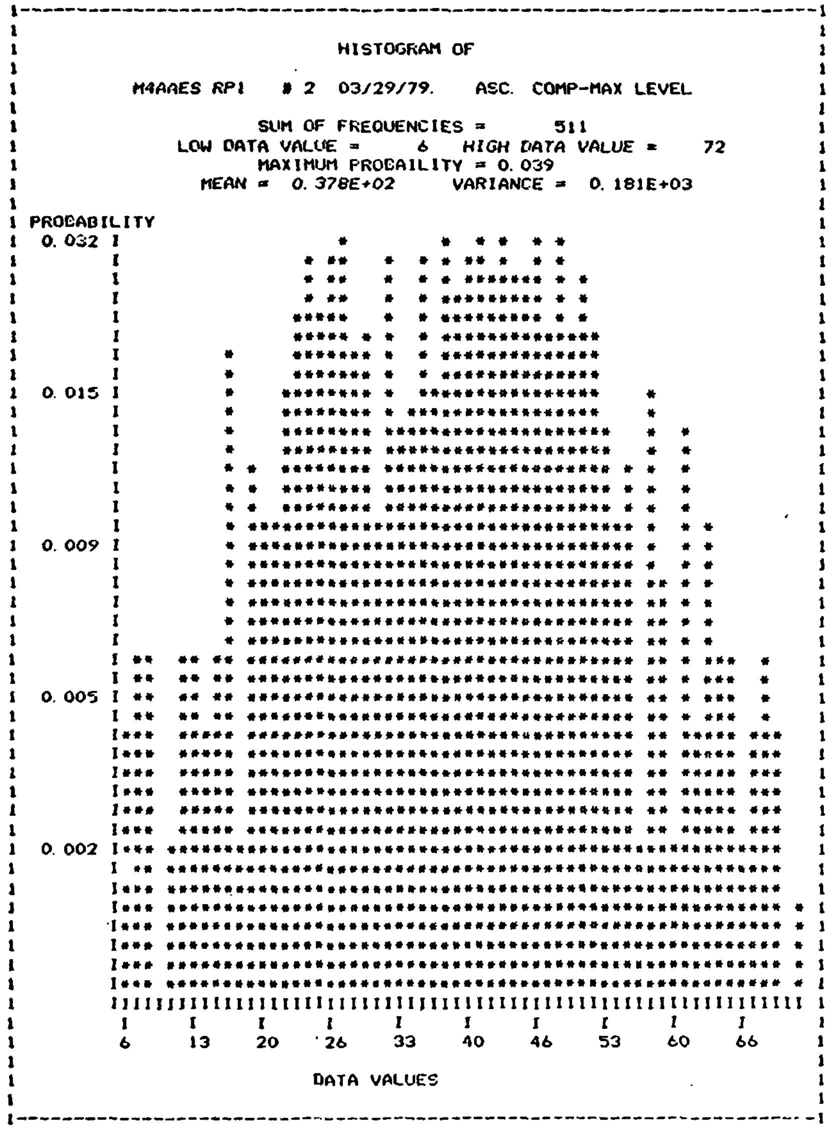


```
                 HISTOGRAM OF

M4AAES RP1   # 1  03/29/79.      ASC. COMP-SIZE

           SUM OF FREQUENCIES =       511
     LOW DATA VALUE =      1   HIGH DATA VALUE =   136
           MAXIMUM PROBAILITY = 0.183
        MEAN =  0.837E+01      VARIANCE =  0.152E+03
```

Figure 5.2.3h. Histograms of the sizes for the ascending components of the minima of M4A. Shown in log and regular scale. Sizes are in number of pixels.



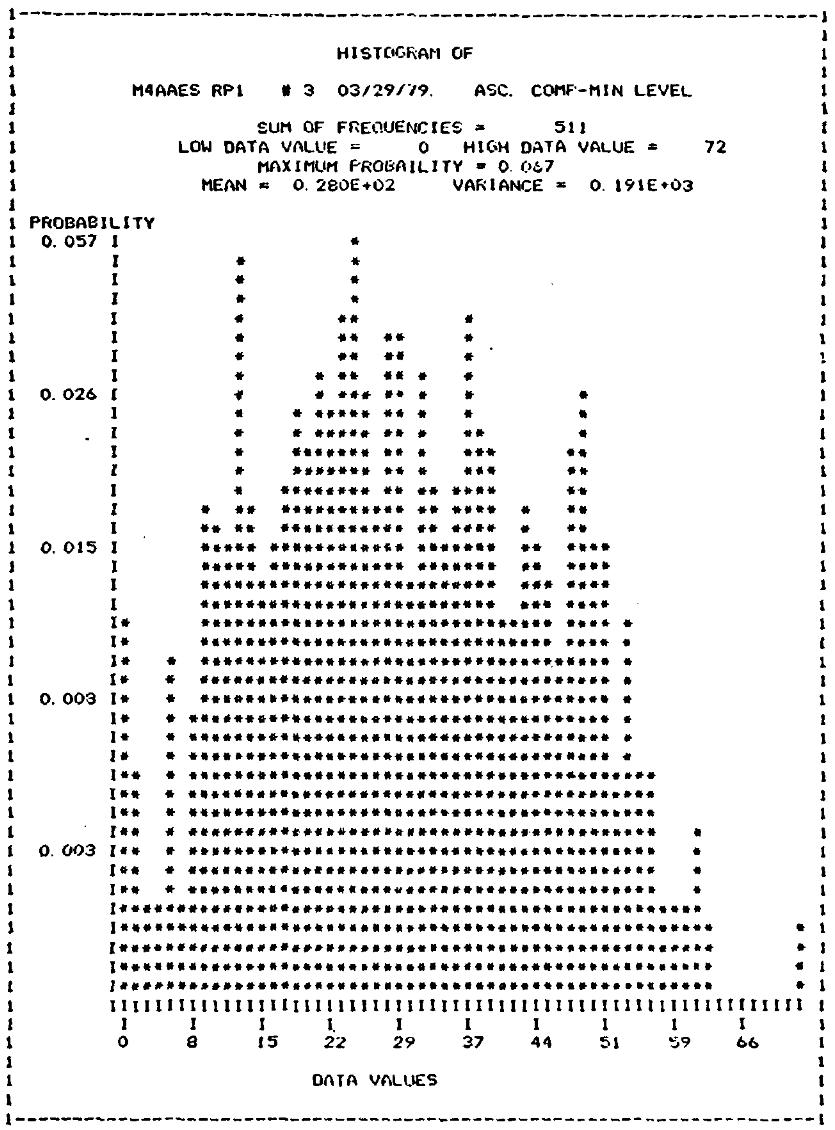
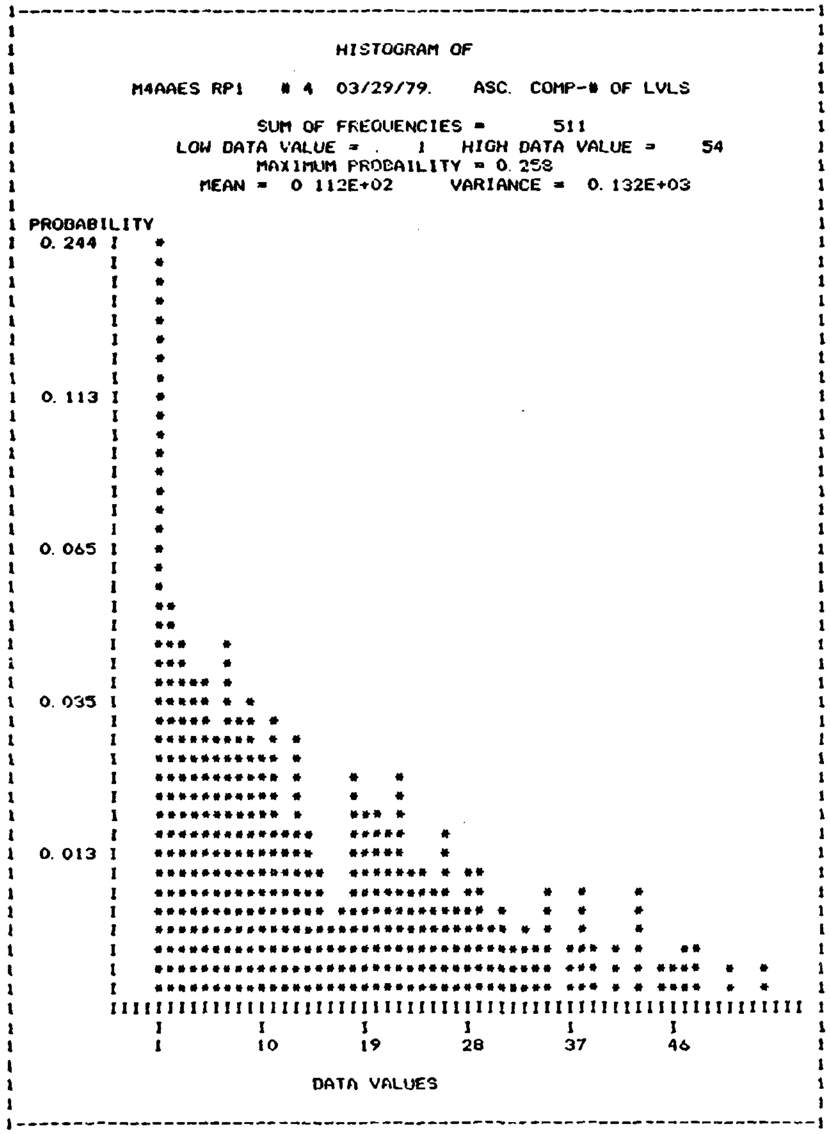


Figure 5.2.3i. *Histograms of the maximum and minimum grey levels for the ascending components of the minima of M4A.*



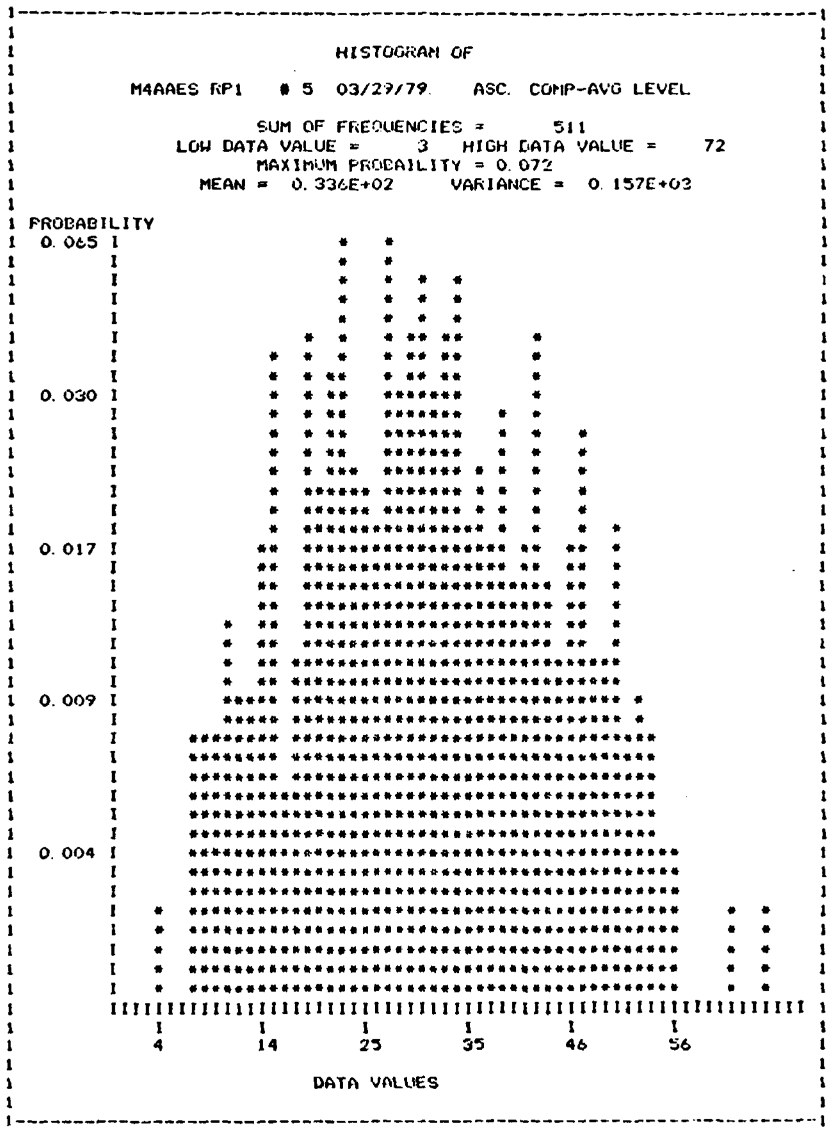
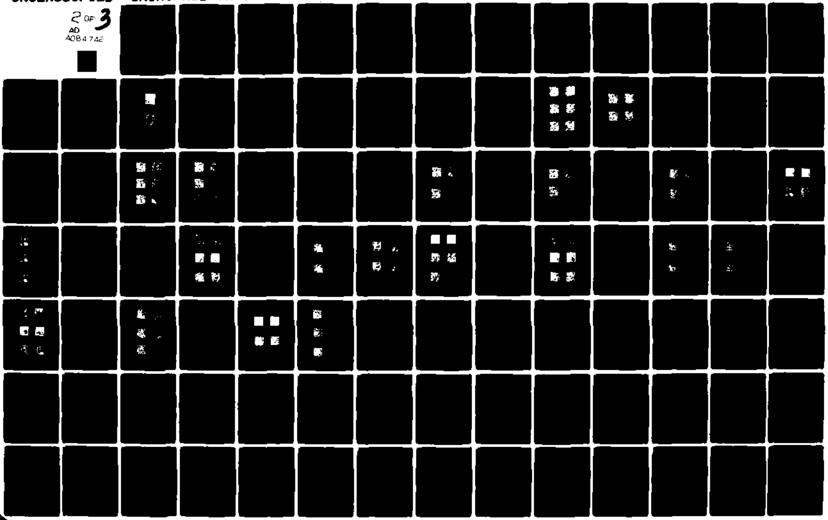


Figure 5.2.3j. Histograms of the number of levels (relative height/depth) and the average grey level (average depth) for the *ascending components of the minima of M4A.*

AD-A084 742 KANSAS UNIV/CENTER FOR RESEARCH INC LAWRENCE REMOTE --ETC F/G 9/2
INVESTIGATION OF EXTREMA IN DIGITAL IMAGES FOR TEXTURE ANALYSIS--ETC(U)
MAR 79 A SINGH, R M HARALICK DAAK70-77-C-0156
UNCLASSIFIED CRINC/RSL-TR-357-1 ETL-0210 NL

2 OF 3
AD
A084742



"measurement vector" or "measurement pattern" or "spectral signature" of the primitive. Because of this the clustering operator belongs to the class of measurement space operators as it operates on the measurement space defined by the data ranges of attribute values.

Inherent to any clustering operation is the distance or similarity measure used. This presented a problem as the components in the measurement vector corresponded to values of different attributes. Arithmetic operations could not be used directly to compute say the Euclidean distance measure. In order to preserve the individuality of each attribute, the distance used was the Mahalanobis distance for two vectors based on a set of vectors. This is defined below.

Let T be a finite set of S measurement vectors $\{u_1, u_2, \ldots, u_S\}$. Each vector u_s is of the form $u_s = \langle u_s^1, u_s^2, \ldots, u_s^K \rangle$, where K is the number of components of each vector. Also associated with each vector is a frequency or weight v_s. In our case this will be the size of the primitive region.

For each of the K components we first compute the average value over the S vectors. That is compute:

$$\mu^k = \frac{1}{S} \sum_{s=1}^{S} u_s^k, \text{ for } k = 1,2,\ldots,K$$

The mean vector μ is then given by $\mu = \langle \mu^1, \mu^2, \ldots, \mu^k \rangle$. Next we compute the covariance matrix as the direct product between (K x 1) column vector $(u_s - \mu)$ and its transpose $(u_s - \mu)'$, which is a (1 x K) row matrix. It is a K x K matrix.

Definition 5.3.1

Let u_s, μ, and S be as given above. The covariance matrix Σ is given by:

$$\Sigma = \frac{1}{S} \sum_{s=1}^{S} (u_s - \mu)(u_s - \mu)'$$

The entire product is divided by the number of vectors S for normalization. The Mahalanobis distance function between two vectors of this set is given next.

Definition 5.3.2

Let T be the set of measurement vectors as given above. The Mahalanobis distance function between two vectors u_i and u_j of this set, denoted by $d(u_i,u_j)$ or just d_{ij} is given by:

$$d(u_i,u_j) = (u_i - u_j)' \Sigma^{-1} (u_i - u_j)$$

where

Σ^{-1} is the inverse of the covariance matrix of Definition 5.3.1

This gives a distance measure between two vectors at the same time maintaining the individuality of each attribute. The limitation is that its data dependent. If we add a few more vectors to the set, distance values between vectors will change as the covariance matrix is changed.

The clustering procedure is an iterative one. In the first iteration we consider each region as a cluster. Each iteration groups together the clusters obtained in the previous iteration. At the same time, it generates measurement vectors for the current set of clusters to be used for the next iteration. A new measurement or signature that is generated for a group is a function of the measurement vectors of the clusters that were put together to form that group.

In generating this new measurement vector, it is useful to use a weighting factor for this computation. For example, if primitives x and y are put together in one group and one of the attribute-components was the height h, then we need to define the height attribute value for the group. This can be $(h_x + h_y)/2$. However, if the two primitives differ in size considerably its better to compute this new height value as $(v_x h_x + v_y h_y)/(v_x + v_y)$, where v_x and v_y are the sizes of the two primitives which we use as weights. The size for the group will be $(v_x + v_y)$. However, this weighting may not be desirable for all the attributes. For example, for the orientation attribute we may want the orientation to be the average orientation regardless of the size of the regions. In the implementation of the clustering algorithm it was possible to indicate which attributes should be weighted and which should not.

We now look at a clustering scheme for a set of measurements patterns. This method is called "Orbit clustering", owing to the similarity in definition of the clusters with that of orbits of states in automata theory [Bavel (1968), Chapter 7]. We describe the process first in a purely mathematical setting to emphasize that it is a pure clustering operator, and not restricted to image category clustering.

We are interested in finding for each vector the vector closest to it in T. Owing to the finiteness of T we are always guaranteed of finding one, no matter how far away it may be. However, if the distance of a vector to its closest neighbors is too much, we want to treat that vector as isolated. This idea can be expressed by the following function.

Definition 5.3.3

Let T be as described above. A function $f{:}T \to T$ is called the Nearest Neighbor function and is defined by:

$$f(u_i) = u_j \text{ where } \lambda > d_{ij} = \min_{1 \leq s \leq S} \{d_{is}\}$$

$$= u_i \text{ otherwise}$$

In the case of two or more vectors being equidistant from a vector, anyone is chosen arbitrarily. The exact formulation of λ is not important right now. It is sufficient to realize that it provides a check to ensure that we do not group together vectors which are too far away.

Definition 5.3.4

Let T and t be as given above. The Orbit of a vector, denoted by O(u), is given by:

$O(u) = \{f^n(u) | n \in ¶\}$, where ¶ is the set of non-negative integers, and

$$f^n(u) = f(f^{n-1}(u))$$

The orbit of a vector "u" is a set of vectors. The vector u is called a generator of O(u).

Before we can see how we can use the notion of orbits in clustering, some results have to be established. Clearly, in order to generate O(u), we should not have to look at f composed with itself, infinite number of times, as the definition suggests. The lemma below shows that that is indeed true. To generate O(u), the function f has to be composed of a finite integer number of times. This is a direct consequence of the finiteness of T. In extreme cases O(u) may consist of u alone or may encompass T entirely.

Lemma 5.3.1

Let $T = \{u_1, u_2, \ldots, u_S\}$ be a finite set of measurement vectors. Then the orbit of vector u is given by:

$O(u) = \{f^n(u) \mid 1 \leq n \leq m\}$ where $m \in ¶$

Proof

The orbit of a vector is found by successive applications of the function f. Each application generates another vector. Suppose that as we apply f, we mark off the different $r \in T$ as being included in O(u). We keep track to see if f generates any vector which is already included. Two possibilities can occur.

Case 1 -- all the elements of T are marked as being in O(u), without encountering any of these twice. In this case exactly (S - 1) application of f must have been made, where S is the cardinality of T.

In the (S - 1) + 1 = S application we will generate a vector which is already a part of O(u), and successive applications will also generate vectors already included in O(u). Thus, O(u) = T and m = S - 1 here.

Case 2 -- suppose in the k^{th} application, k < S, we encounter a vector which is already included in O(u), i.e., $f^k(u) \in O(u)$. Then:

$f(f^k(u)) = f^{k+1}(u) \in O(u)$, for $f(f^k(u))$

will generate the same vector, the first time $f^k(u)$ was encountered, and that has already been included in O(u). Similarly, $f(f(f^k(u))) = f^{k+2}(u)$ will also give a vector already included in O(u). In general $f^{k+j}(u)$, $j \in ¶$ will be vectors which have been included in O(u) previously. Thus, further generations will not be necessary and m = k - 1.

In each case m is a finite number and gives an upperbound to the number of compositions of f.

f is a function that generates the next vector in the sequence of producing the orbits. We now need a function which goes the other way. This is a function that when applied to $u \in T$ gives us the set of all predecessor vectors of u. A vector u_j is called a predecessor vector for u_i, if and only if there exists a $n \in ¶$ such that $f^n(u_j) = u_i$. Actually it is more convenient to define this predecessor function as a set function, rather than a point function. This function is denoted by "g", and its domain and range sets are the power set of T. The definition follows.

Definition 5.3.5

Let T be a set of measurement vectors as described previously. Let $R \subseteq T$. The Predecessor function is given by $g: \mathcal{P}(T) \rightarrow \mathcal{P}(T)$. The Predecessor set of R, denoted by g(R), is defined by:

$$g(r) = \{u \in T \mid f^n(u) \in R, n \in ¶\}$$

and

$$g(\Phi) = \Phi$$

The definition states by applying f sufficient number of times to $u \in g(R)$ we generate a member of R.

The clusters that the Orbit clustering operator creates can be expressed using the function g. Simply, the clusters generated in one iteration from the set of vectors T is the set:

$$\{g(O(u_i)) \mid 1 \leq i \leq S\}, \text{ where } S = \#T$$

A cluster is determined by generating the orbit of a state and then taking its predecessor set. It remains to show that these clusters are well defined. Each vector is assigned to one and only one cluster. That each vector is assigned to a cluster is evident from the definition of the set of clusters. The fact that it can only appear in one cluster is given by the lemma below.

In the following we will refer to a vector $u_i \in T$ by its index "i". Thus, $O(u_i)$ becomes $O(i)$. This is only done for notational simplicity. We will also make use of the fact that the composition of f with itself is commutative. For two integers a and b we have:

$$\begin{aligned} f^a(f^b(u)) &= f^{a+b}(u) && \text{from definition of } f \\ &= f^{b+a}(u) && \text{by commutativity of addition} \\ &= f^b(f^a(u)). \end{aligned}$$

<u>Lemma 5.3.2</u>

Let T be a set of vectors denoted by $\{1,2,\ldots,S\}$. Then for all i, $j \in T$ either:

$$g(O(i)) \cap g(O(j)) = \Phi$$

or

$$g(O(i)) = g(O(j)).$$

<u>Proof</u>

Suppose $g(O(i)) \cap g(O(j)) \neq \Phi$. We need to show then that:

$$g(O(i)) = g(O(j))$$

Now $g(O(i)) \cap g(O(j)) \neq \Phi$ implies that there exists $k \in g(O(i))$ and $g(O(j))$. Further, $k \in g(O(i))$ implies $f^{n_i}(k) \in O(j)$, for some $n_i \in ¶$ and $k \in g(O(j))$ implies $f^{n_j}(k) \in O(j)$, for some $n_j \in ¶$.

Let $p \in g(O(i))$. To show that $p \in g(O(j))$:

$$p \in g(O(i)) \text{ implies } f^{n_1}(p) \in O(i), \text{ for some } n_1 \in ¶$$

Since both $f^{n_i}(k)$ and $f^{n_1}(p)$ are members of $O(i)$, then at least one of them must generate the other by successive application of f. It is possible that both can generate each other, but that does not matter.

<u>Case 1</u> -- if $f^{n_i}(k)$ generates $f^{n_1}(p)$, then

$f^{n_2}(f^{n_i}(k)) = f^{n_1}(p)$, for some $n_2 \varepsilon \P$

Applying f^{n_j} to both sides we have

$$\begin{aligned} f^{n_j}(f^{n_1}(p)) &= f^{n_j}(f^{n_2}(f^{n_i}(k))) \\ &= f^{n_2}(f^{n_j}(f^{n_i}(k))) \\ &= f^{n_2}(f^{n_i}(f^{n_j}(k))) \end{aligned}$$

But $f^{n_j}(k) \varepsilon O(j)$.

Therefore, $f^{n_2 + n_i}(f^{n_j}(k)) \varepsilon O(j)$ by definition of $O(j)$: i.e.,

$f^{n_j}(f^{n_1}(p)) \varepsilon O(j)$, which implies $p \varepsilon g(O(j))$.

<u>Case 2</u> -- if f^{n_1} generates $f^{n_i}(k)$, then $f^{n_3}(f^{n_1}(p)) = f^{n_i}(k)$, for some $n_3 \varepsilon \P$. Applying f^{n_j} to both sides we have:

$$\begin{aligned} f^{n_j}(f^{n_3}(f^{n_1}(p))) &= f^{n_j}(f^{n_i}(k)) \\ &= f^{n_i}(f^{n_j}(k)) \varepsilon O(j), \text{ since } f^{n_j}(k) \varepsilon O(j). \end{aligned}$$

Therefore, $f^{n_j + n_3 + n_1}(p) \varepsilon O(j)$ implies $p \varepsilon g(O(j))$.

In either case, $p \varepsilon g(O(j))$. But p was chosen arbitrarily in $g(O(i))$. Therefore, $g(O(i)) \subseteq g(O(j))$.

By a symmetric argument it can be shown that $g(O(j)) \subseteq g(O(i))$. Hence, $g(O(i)) = g(O(j))$.

The lemma says that if two clusters have one or more vectors in common, then they must be the same cluster. Thus, each vector is assigned to one and only one cluster, and the clusters are well defined.

Having obtained the clusters in one iteration, we need to generate the measurement vectors for this new set of clusters. This is done according to the definition below.

Definition 5.3.6

Let the set of vectors which form a cluster be denoted by the indices $\{n_1,n_2,\ldots,n_r\}$. The frequency V and the measurement vector $<U^1,U^2,\ldots,U^K>$ for this cluster is given by:

$$V = \sum_{j=1}^{r} v_{n_j}$$

$$U^k = \left(\sum_{j=1}^{r} v_{n_j}\, u^k_{n_j}\right) \Big/ V, \text{ if weighting was specified or}$$

$$U^k = \left(\sum_{j=1}^{r} u^k_{n_j}\right) \Big/ r, \quad \text{if no weighting was specified.}$$

In summary, one Orbit clustering iteration consists of generating the clusters $g(O(u_i))$, $i = 1,2,\ldots,S$, and the means for the next iteration. This procedure is repeated until the number of clusters is reduced to less than some number desired.

This clustering method has one drawback. It is agglomerative in nature as it only coalesces clusters, but never splits one that may be badly defined. It is important, therefore, that we have some control over which elements are being put together. The parameter λ, mentioned before, is one such regulator. It ensures that if the minimum distance, in metric space between a cluster and its neighbor, is too much, the cluster is not grouped with any other. In that iteration it forms a group of its own.

The implementation of λ can be done in different ways depending on what is desired. For example, one may have some idea of what the distances in measurement space mean. In this case the maximum allowed can be fed into the program before each iteration. Another way is to define λ in a relative manner. This is done by computing the average (μ) and the standard deviation (σ) of the minimum distances. We denote by d_i, the distance to its closest neighbor (minimum distance) for cluster "i".

$$\mu = \left(\sum_{i=1}^{S} d_i\right)/S \quad \text{and} \quad \sigma^2 = \left(\sum_{i=1}^{S} (d_i - \mu)^2\right)/S$$

The cutoff λ is given by $\lambda = \mu + \theta\sigma$, where θ is some real number. Here the cutoff is controlled with respect to the distribution of the data. If σ is very large, then a small θ or even negative value may be entered. This ensures that the cutoff is reasonable. There are other ways of introducing λ. In the implementation of the algorithm only the above two methods were included. For each iteration the user is asked for θ and the absolute maximum distance allowed. The actual cutoff chosen is the minimum of the two. This gives a little additional control.

The use of the clustering process on image data can be described as follows. First a table is generated which contains the size and attributes of the primitives we wish to cluster on. These are obtained by the property extraction scheme described previously. For each attribute it is also indicated if weighting is to be used or not in computing group measurement vectors. The table gives the initial set of measurement vectors and frequencies for the first iteration of the Orbit clustering operator. The operator is applied repeatedly until the number of clusters have been reduced to what is desirable.

During the clustering, we keep track of which cluster each of the original regions belongs to. The primitive image containing these regions is then relabelled according to the cluster codes. This gives us the clustered image.

We mentioned before that one drawback of this method was that it only coalesced clusters and never checked to see if any needed to be split. This, however, does have an advantage in an indirect and practical way. The execution time is reduced considerably, but even more, the core requirements are minimal. Most clustering procedures which group and split clusters require memory in order of S^2, where 'S' is the number of elements being clustered. This scheme requires order of S. That is very useful for large data sets. The execution time is still order of S^2 per iteration, because for each vector the rest of the vectors have to be examined to find which one is closest to it.

The reader may have noticed some similarity between the orbit clustering described above and the method of single linkage clustering [Sneath and Sokal, 1973; Anderson, 1973; Hartigan, 1975]. This similarity exists only for the first iteration. Being agglomerative methods, both begin with the assginment of one case per cluster, and then looking for nearest neighbor links. In general, in the single linkage scheme, two clusters may be merged if any of their members lie close enough. That is not so for the Orbit clustering method. There the links are computed between cluster centroids and not between pairs of individual members of the clusters. The centroid, which is computed using the weighted means of the members of the cluster, is a much truer representative of the cluster's position in measurement space. Links based on centroid coordinates are, therefore, a better critiera for merging clusters. Their use also reduces the problem of "chaining" which tends to occur when using single linkeage. Furthermore, the use of control parameters ensures the merging of only those clusters which lie close enough to each other. Single linkage clustering procedures do not include these regulators.

In any iteration, it is not possible to specify the number of clusters desired. That is really a function of the data set and the thresholds entered. For this very reason, the process is sometimes referred to as unsupervised clustering. In supervised clustering schemes, the number and kinds of clusters are determined before the clustering takes place. This requires some prior knowledge about the data. In image category clustering, the information is usually entered as ground truth data. Without it, it is difficult to perform any supervised clustering. In the unsupervised case, no prior information is necessary. Only after the result is generated, does one sit down for analysis with the ground truth.

The clustering scheme was applied to the various to the extrema and reachability set primitives. The results of this and post processing are discussed in the next chapter. To close this one, the last section looks at attribute values from the strong texture measure point of view.

5.4 Strong Measures

The previous sections described the weak texture measures as we were mostly looking at the distribution primitives and their attributes singly. More complex measures would involve a definition of the spatial relationship between these primitives. This was not part of the research carried out, but is included here as a natural extension of the primitive-attribute values for stronger measures.

Including spatial relationships gives rise to second order counts and co-occurrence. In the case of the pixel primitive and its gray tone attribute, spatial relationship was distance between pixels. These resulted in the gray tone co-occurrence counts. This idea may be extended to any primitive. Co-occurrence of primitives with attribute values gives rise to the concept of the Generalized Co-occurrence Matrix (GCM) first investigated by Davis et al. (1979).

The first step is to decompose an image into its primitives which we denote by the set Q. Let T denote the set of primitive properties such as size, shape, mean gray tone, etc. Also let f be a function assigning to each primitive in Q a property of T. Finally, let $S \subseteq Q \times Q$ be a spatial relation paring all primitives which satisfy the spatial constraint. The Generalized Co-occurrence Matrix (GCM) P is given by:

$$P(t_1,t_2) = \frac{\#\{(q_1,q_2) \in S \mid f(q_1) = t_1 \text{ and } f(q_2) = t_2\}}{\#S}$$

$P(t_1,t_2)$ is the relative frequency with which two primitives co-occur in the specified spatial relationship, one having the property t_1 and the other having the property t_2.

The primitives Davis et al. used were edge detectors with directional information. The spatial relationship was in form of a general constraint predicate which measured spatial proximity by two primitives being within a distance "k" or a primitive being the nearest neighbor of the other. These predicates could be combined with orientation constraints as well to emphasize directionality if needed. It should be noted that in the spatial relation using nearest neighbor predicate, the GCM is no longer symmetric.

Davis et al. compared GCM's and the gray tone co-occurrence matrices on the same set of images and obtained much better discrimination with the former.

The computation of the GCM is a much more difficult task in the general case, especially if the spatial relation is complex. Davis et al. simplified it by making their primitive size that of a pixel, and counting pixels that co-occurred in the spatial relationship. The same thing may be done if the primitives were extrema or ascending/descending components. Each could be represented by a single point, preferably their center of gravity. Counting could then be done by laying a window around each pixel. This would guarantee that no primitive pair would be counted twice. However, it would be nice to have a general program which would count proximity for regions of arbitrary size. This would be complex as one would have to keep track of which pairs of primitives had been counted and which had not.

Another way to solve the counting problem is to define spatial relationships as adjacency. The method involves growing the primitive regions out till they touch each other. From this image a region adjacency graph (RAG) denoting which primitives are adjacent can be generated and counting can be carried out on it. Figure 5.4.1 shows such a filled image. This was the result of growing the min-max extrema till they touched each other.

A brief note here about the terminology weak and strong measures. We have been characterizing first- and second-order statistics by the above two terms. Actually the difference between their performance may not be as wide as the terms strong and weak suggest. They were chosen here for want of better ones.

"Weak" measures are not really that weak. As we have seen, a lot of texture information for discrimination and segmentation can be extracted from these statistics. The question to ask is whether there is anything that can be obtained by second-order measures that could not be extracted by first-order methods? Putting it another way, given a second-order statistic,

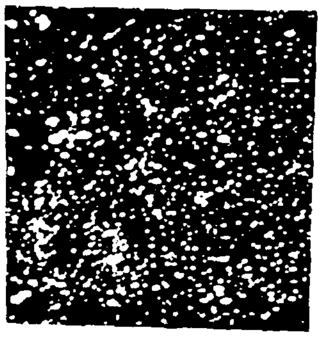


Figure 5.4.1a. Uniquely labelled extrema of M4A.

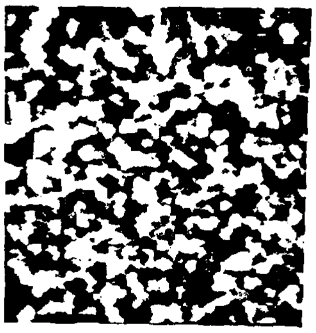


Figure 5.4.1b. Extrema labels grown out to determine adjacency.

is there a set of first-order statistics which used in combination with each other or iteratively, that could perform the same task? As an example, let us look at the problem of discriminating between the two images of Figure 2.1.1. We stated that based on histograms these few images could not be distinguished, but co-occurrence matrices could separate them. However, there are other first-order measurements that can. An example is to compute the root mean square (RMS) error between the images. The RMS error, E, is defined as follows for two images I and J of the same dimensionality, N_r row by N_c columns:

$$E = \left[\frac{\sum_{(i,j) \in Z_r \times Z_c} (I(i,j) - J(i,j))^2}{N_r \times N_c} \right]^{1/2}$$

This is a point by point comparison and uses no spatial information. If the error is large compared to the gray tone range, the two images would be considered distinguishable.

Another simple weak operation to distinguish them would be to run a 2 x 2 box filter and then look at the histograms. As seen from Figure 5.4.2 the resulting histograms are quite different and thus can distinguish the image. The images are smaller as we lose one row and column owing to edge effects in the filtering operation.

At this time there is no clear answer to the question posed above.

1	1	3	3
1	1	3	3
1	1	3	3
1	1	3	3

Image A

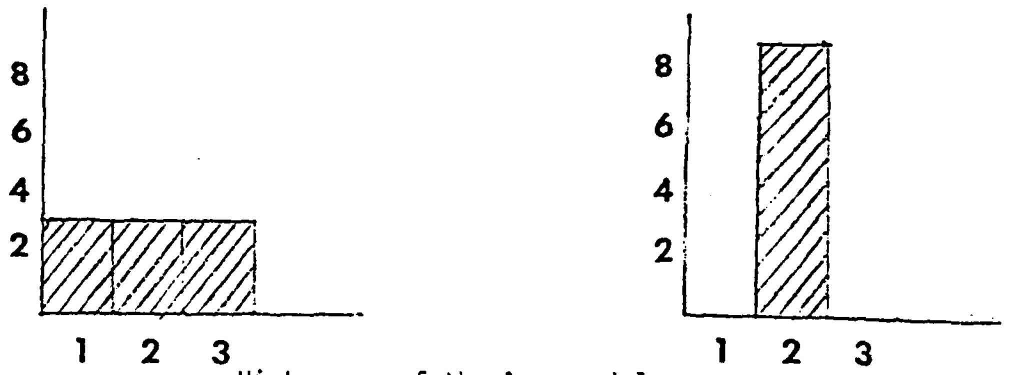
1	3	1	3
3	1	3	1
1	3	1	3
3	1	3	1

Image B

1	2	3
1	2	3
1	2	3

2	2	2
2	2	2
2	2	2

Images After a 2 x 2 Average Filter



Histograms of the Averaged Images

Figure 5.4.2. Effect of a 2 x 2 average filter on images of Figure 2.1.1.

6.0 CLUSTERING RESULTS

We now look at the results of applying the clustering scheme described in the last chapter. The clustering was carried out on both the photographic and radar bands of the images provided by ETL. Since it would be impractical to describe all the processing that was done, we will look at some results of processing four subsections of the ETL imagery. These subsections were selected for their representative textures. One of these is the image M4A encountered previously. This and other images are discussed in the sections to follow.

Subsections of the original 512 x 512 images were chosen instead of the entire image owing to computer limitations. The development of the software and processing was done on a PDP-15/20 with the RSX Multi-Access operating system. While this system has a lot of flexibilities, it severely restricts the size of the image data files on the disc. As a result the largest image that could be processed was about 200 x 200. Most of the images selected however were 128 x 128, as this gave the optimum performance in disc storage and processing times. The algorithms developed though are quite general and can easily process larger images if hardware limitations are removed.

6.1 Attributes for Clustering

As mentioned before the clustering operator is not restricted to image data but was set up to cluster any set of cases for which an attribute value table was available. Here the cases were the primitives - maxima, ascending components etc., and the attribute value table was the property values for the various attributes discussed in Section 5.1. Thus the clustering process first involved a generation of an attribute value table for the primitives in question, by the property extraction operators. The table contained a property value for each attribute specified for each primitive. Clustering was then performed on this table. The resulting cluster code list was used to create the clustered image from the corresponding uniquely labelled image of the primitive regions. This separation of the clustering operation from tne image domain was done to keep the clustering general, as well as to maximize core work area during processing.

Of the properties discussed in Section 5.1, five were selected and used extensively for all the images. These five were:

1) Size of the primitive,
2) Maximum gray level,
3) Minimum gray level,
4) Number of levels in the primitive and
5) Average gray level.

The size of the primitive is the number of pixels in the primitive. This information was always required by the clustering operator, whether or not it was specified as a property to be used. As given in Definition 5.3.6 this size corresponds to frequency and was necessary for computing the measurement vectors for the new groups, for the next clustering iteration, if weighting was specified for a particular attribute. In the actual implementation of the code, two types of weighting schemes were allowed. To compute the attribute value for a set of cases in a group, one could either weigh by the pixel sizes or by the number of regions comprising each case. If the region sizes were about the same, the latter option seemed more feasible. Essentially the second option amounts to assigning a weight of '1' to each row of the initial attribute value table. Weighting by number of regions was specified for all properties except the region size itself, as that would have been meaningless. No weighting option was specified for the size property.

The maximum gray level corresponds to the absolute height for the maxima and descending components (valley height for ascending components). The minimum gray level corresponds to peak descent for the descending components (valley depth for the minima and ascending components). The number of levels is the range (= Maximum-Minimum+1) of levels for the descending and ascending components. It gives a measure of the relative height of a peak or valley. The average level is the average graytone over the primitive region.

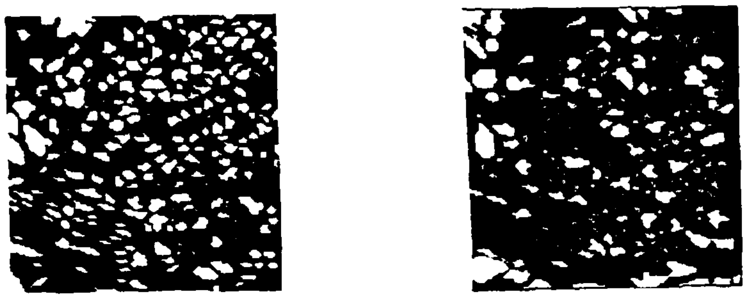
For the extrema primitives only the maximum gray level is meaningful. The others are redundant or of little value. For example the number of levels for an extremum primitive is always one, and the minimum level is always the same as the maximum level. The sizes for most extrema is one with very little spread. The five attributes are more meaningful for the reachability sets. Also the distributions of these attributes are more spread out as can be seen from the histograms of Figure 5.2.2. This would allow for better clustering. Since the maximum/minimum gray level for the extrema was the same as for the corresponding reachability sets, only the latter primitives were used in the clustering experiments. Not only did they include all the meaningful properties of the extrema, but owing to their larger size, they gave a better spatial definition to the different textured regions of the image.

To get a visual idea of how well these attributes characterised regions of different texture, property images were generated for the reachability sets and the five attributes mentioned above. Along with the histograms these images helped suggest which properties might be more useful for clustering.

The property images were generated the same way as described in Section 5.2. Figure 6.1.1 shows these images. The maximum level (property 2) looked the best for both reachability sets followed by the average level (property 5) and the minimum level (property 3). The number of levels (property 4) seemed very mixed and did not suggest direct segmentation. It could perhaps have been more useful as a secondary component in the measurement space clustering. The size property (property 1) attribute was a little more consistent but not very illuminating by itself, except to bring out some large flat no-texture regions.

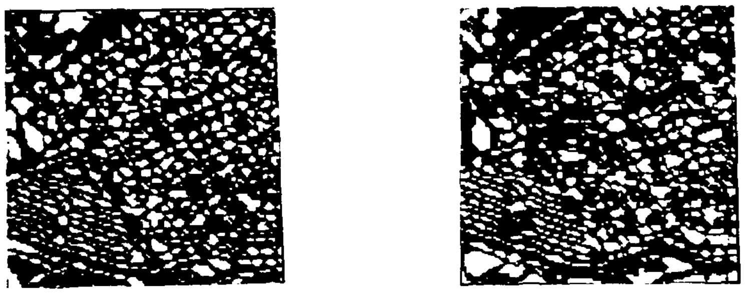
The clustering was carried out using different combinations of the five attributes mentioned above. Examples of these are given in the following sections. It sould be noted that property 4, the number of levels in a primitive region, could not be used with both the maximum and minimum gray levels. This is because it is a linear combination of the two. Using all three attributes together would make the covariance matrix singular. Thus the combinations of attributes that were allowed was restricted slightly.

Before discussing the processing on the individual images there are two points that remain to be covered. The first deals with the level of clustering. As mentioned in Section 5.3 the clustering operator is



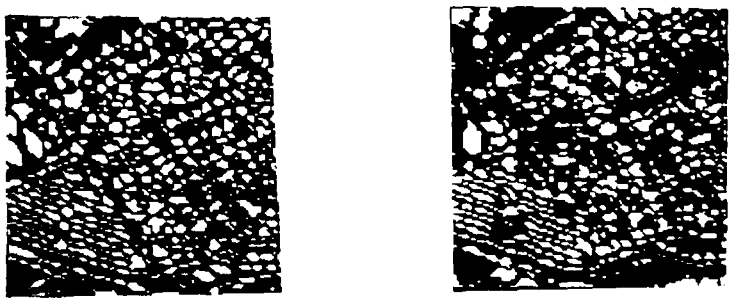
Descending Components   Ascending Components

Property 1 - Size of primitive regions.



Descending Components   Ascending Components

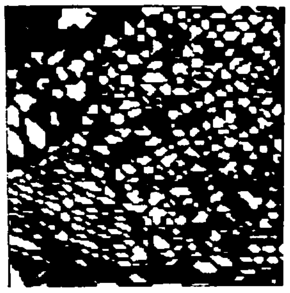
Property 2 - Maximum gray level.



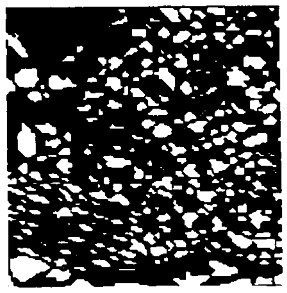
Descending Components   Ascending Components

Property 3 - Minimum gray level.

Figure 6.1.1. Property image for the Descending and Ascending components of M4A. Similar regions indicate regions with similar property values.

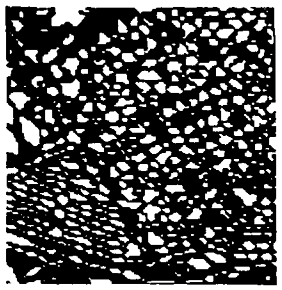


Descending Components

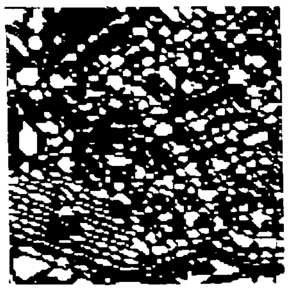


Ascending Components

Property 4 - Number of gray levels.



Descending Components



Ascending Components

Property 5 - Average gray level.

Figure 6.1.1. (continued)

a coalescing one. In each iteration the number of groups decreases or remains the same. The latter occurs if the user-entered thresholds are too restrictive. A problem with this and for that matter all non-supervised clustering algorithms, is to know when to stop. In a supervised environment the number of final clusters is usually known or specified at the beginning. Here without extensive ground truth it becomes in part a guessing game. The problem is further compounded by the noise in the image. Even if we were to know that there are five classes in the image, it does not follow that the non-supervised clustering should yield exactly five clusters. Usually we would have to settle for more than five groups as some clusters may be noise regions of the image. The best way to solve this problem is to examine images for the final four or five stages of the clustering. By then the number of clusters has been reduced to a managable range and each can be examined individually. An example set of images is given in the next section. In our processing, along with the examination of the clustered image, a table of cluster groupings with measurement vectors was also printed and analysed, for each iteration. Based on these, the best stopping point was determined. The cluster groupings indicated how many original regions there were in each group. Thus if in a late stage in the clustering, one group consisted of only one region, then it was either a noise region or one with an extreme attribute value. An examination of the measurement vectors printout would clarify this. Instances of this occured in many images as there was usually atleast one flat region with an extreme size value.

The second point deals with the spatial generalization operation on the clustered image. This is sometimes also referred to as region growing. The idea here is to incorporate the unclassified part of the image into the segments determined by the clustering. The unclassified area consists of pixels which did not fall into the unique reachability sets. The premise behind the spatial generalization is that the pixels in the unclassified area adjacent to the primitive region, very likely belong to the segment that the region has been assigned to. This is especially true of unclassified pixels which are surrounded by primitive regions of the same texture class. By growing these regions out, the unclassified pixels are included and a segmentation of the entire image is achieved. The growing is an iterative process, each iteration being one scan of the image. In succesive iterations the regions grow one pixel at time, until they meet each other or the picture edge. The growing is terminated when the whole image is filled or by fixing the number of iterations. The definition below summarises this.

<u>Definition 6.1.1</u>

Let $T = \{ I | I:Zr \times Zc \rightarrow L \}$ and $T' = \{ I' | I':Zr \times Zc \rightarrow L \}$ be a set of domain and range images. L is a set of labels and let N(i,j) denote a neighborhood for cell (i,j). An image operator $GR:T \rightarrow T'$ is a region growing operator if $GR(I) = I'$ and

a) If $I(i,j) \neq 0$, then $I'(i,j) = I(i,j)$
b) If $I(i,j) = 0$, then $I'(i,j) = 0$ if $\#\{(k,l) | (k,l) \in N(i,j) \ \& \ I(k,l) \neq 0\} = 0$
else $I'(i,j) = I(k,l)$ for $(k,l) \in N(i,j) \ \& \ I(k,l) \neq 0$.

The above is for one iteration of the operator with '0' representing the unlabeled class. Statement (a) says that pixels that are labeled are left labeled. The first part of (b) states that an isolated unlabeled pixel is left unlabeled. A pixel is only labeled if one of the adjacent pixels is labeled. The last part of (b) states that an unlabeled pixel is assigned the class of one of its labeled neighbors.

The neighborhoods that are used are the four and eight neighborhoods of a pixel. These are used alternatingly to ensure an isotropic growth. For most images only a few iterations were needed as the primitive regions were distributed quite evenly. The region growing can result in erratic growth if the initial regions are far apart. In this case the number of iterations should be fixed.

A complementary operator is the region shrinking operator. This takes pixels on the edges of regions and marks them as unclassified, i.e. it 'shrinks' the regions. The purpose of this is two fold. Firstly it automatically gives us the boundaries on a completely filled image. Also when used alternately with the region growing operator it rounds out region boundaries and tends to eliminate small noise type regions. The resulting image is more homogenous looking. See Section 6.2 for examples. The definition for this operator is given below.

Definition 6.1.2

Let T, T', N and L be as in the previous definition. An image operator $SR: T \rightarrow T'$ is called a Region Shrinking operator if $SR(I) = I'$ and

a) If $I(i,j) = 0$, then $I'(i,j) = 0$
b) if $I(i,j) \neq 0$, then $I'(i,j) = 0$ if $I(m,n) \neq I(i,j)$ for some $(m,n) \in N(i,j)$,
$= I(i,j)$ otherwise.

Again the neighborhood here can be the four or the eight neighborhood. In the examples presented the eight neighborhood was used. The pixel is 'shrunk' if any of its neighbors differs from itself.

6.2 Processing Image 'M4A'

Since it was the first to be processed a lot of work was done on this image. The work was carried out to fine tune the parameters for the clustering operation and also to determine which attributes gave better results.

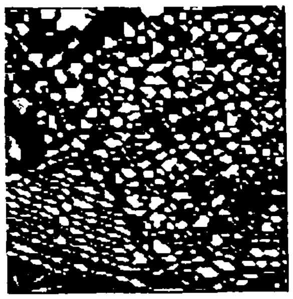
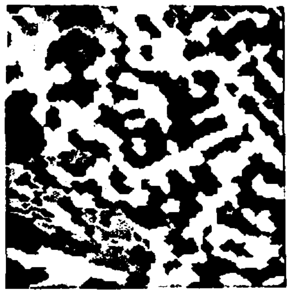
As described in Section 3.4, M4A is a 128 x 128 subimage of the fourth ETL image. Geographically it is a small section of Union City, north of Newark, a suburb of San Francisco. The area contains a trailer court at the bottom left and residential houses with trees over most of the picture. At the top left corner are fields, some of which are quite dark. Running below the trailer court is a highway separating the court and an orchard located at the very bottom left. A little left of the center at the bottom is another dark field which is part of the area between the highway and its exit ramp lane (see Figure 3.4.1).

The residential area which covers about two-thirds of the image is not all one homogenous texture. This could be because the houses which were built and developed at different times, had different spacing between them. Compared to the texture of the trailer court it is much coarser. Broadly speaking, the image has four or five texture regions. In the processing to be described below our aim was to get the best definition for these regions.

The analysis was carried out on both the photographic and radar bands. The latter will be discussed later. On the photo band both the descending and the ascending reachability sets were clustered. These images were illustrated back in Figure 3.4.7. Below, the processing on the descending components is presented first.

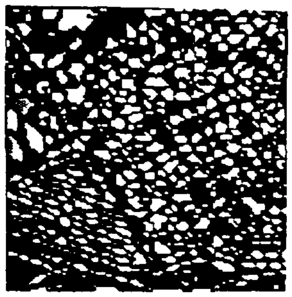
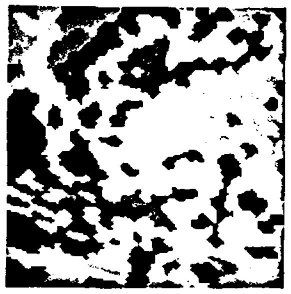
In the example pictures which follow, the different clusters are indicated by different colors. It is not possible in the clustering algorithm to fix a cluster number (code) to any one group, for example the residential area. The codes are assigned to the groups in the order in which they are encountered and the manner in which the algorithm merges them. A region could come out as cluster number two in one processing run or as number four in an another. The colors are consistent in that all regions in the same cluster or class have the same color. Also the mapping of the cluster number to color has been maintained over all the examples. This information is not of much use except to identify the cluster number. There are twenty colors which show twenty classes. If an image has more than twenty classes, cluster codes twenty and higher have the same color. As most images being analysed had around ten classes, this was no problem. The mapping of colors to cluster codes is given in the appendix.

The best result on the descending components was obtained using the size and the maximum height properties (properties 1 and 2). To get an idea of the clustering process, the results of the final five iterations for this run are shown in Figure 6.2.1. The corresponding spatially generalized images are also shown. These images have 25, 19, 16, 11 and 7 classes respectively. In Figure 6.2.1a even though the original 536

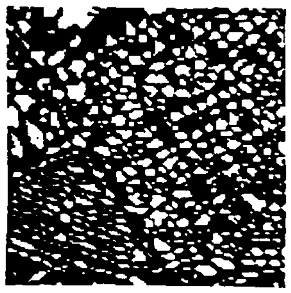
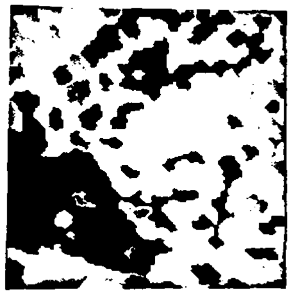
Clustered Image Generalised Image

(a) Twenty five Clusters

Clustered Image Generalised Image

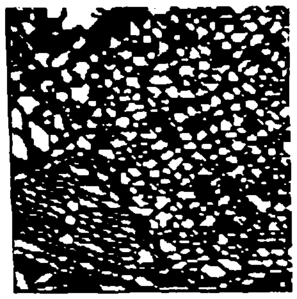
(b) Ninteen Clusters

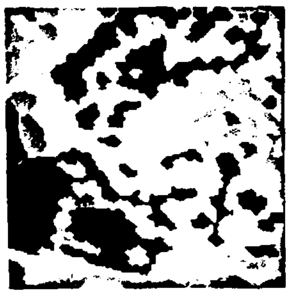
Clustered Image Generalised Image

(c) Sixteen Clusters

Figure 6.2.1. Descending components clustering on photo band of M4A. Properties 1 and 2. Five iterations.

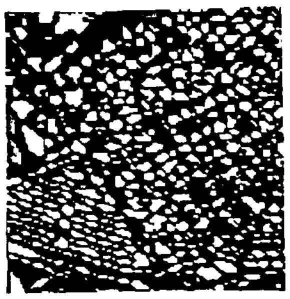


Clustered Image

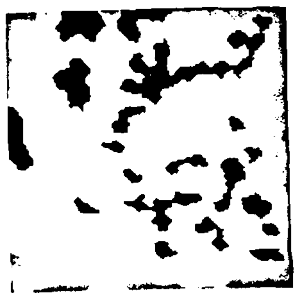


Generalised Image

(d) Eleven Clusters

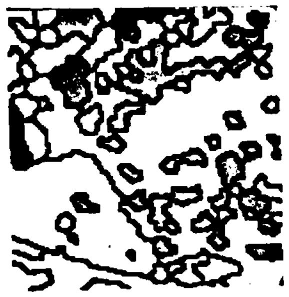


Clustered Image

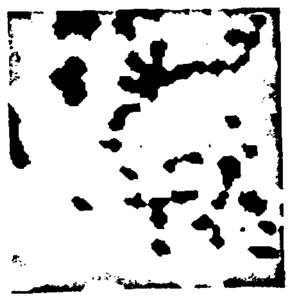


Generalised Image

(e) Seven Clusters



(f) 'Shrunken' image with boundaries.



(g) Smoothed spatially generalised image.

Figure 6.2.1. (continued)

regions have been reduced to 25 clusters, the picture is still quite complex. Further iterations bring out more order. In 6.2.1b the residential area begins to emerge but the trailer court still contains too many segments. By 6.2.1c this has also taken shape and the residential area consists of essentially three classes. In 6.2.1d some of the smaller clusters for the flat fields have begun to emerge, but the residential region is still mixed. Finally in 6.2.1e both the residential and the trailer court clear up to yield the image shown. The discontinuity in the middle of the trailer court is an open ground with perhaps trees. It connects the driveways of the different sections of the court. This was understandably merged with the residential area as it seems to have the same structure.

The residential area still seems to contain two classes and further clustering could perhaps have cleaned this up. However it would have been at the risk of merging other major regions. In Figure 6.2.1f the 'shrunken' image with the boundaries between the different regions is shown. Also in Figure 6.2.1g the spatial generalization of 6.2.1f is presented. On comparison with the filled image of 6.2.1e it can be seen that the process smooths out boundaries and eliminates small mislabeled clusters inside large regions. This is the smoothed spatially generalized image.

The final segmented image of Figure 6.2.1g is far from ideal but is good for segmentation based only on texture, and no other properties of the image. There are some errors. Some of the trailer court primitives had merged quite early on (Figure 6.2.1a) with other primitives in the top center residential area of the image. Once this occurs, it is not possible to separate them. This pattern was observed in many of the runs and an examination of the picture suggests that these primitives are similar in the properties being used and thus not separable. The two part structure of the residential area was hard to eliminate for most all the processing. This is because the texture of this area is much more complex, and again the attributes being used are not enough to capture this complexity. For example we did not use any information regarding regularity of patterns. A glance at the image shows that some houses are arranged in lines along with trees. Using a regularity property we might be able to separate out this patterned texture from the irregular arrangements of houses in other parts of the image.

The results on the flat fields are mixed. Most of them came out quite well but some of them are still split. The reason for this is that we only define the descending components on the bright parts of the image. Thus we only pick up part of the dark fields. The rest of the pixels of the field fall into the non-unique class. Correspondingly for the ascending components we only pick up the darkest parts of the field, with the rest of the field pixels again falling into the non-unique class. This is a basic problem with the reachability sets. The light and dark flat fields result in different spatial definitions when using ascending and descending components. The different reachability sets capture different aspects of fields. One would like to cluster both sets at the same time, but this would involve the use of properties which are common to both, such as size and shape attributes. There are not many of those. The peak heights of descending components cannot be compared directly to the valley depths of ascending components.

Another basic problem with the reachability set approach is that when there is sufficient texture (in terms of number of extrema) they work well. However when the texture is very low or non-existent it becomes difficult for the reachability sets to capture the shape and properties of the area. The result is that we try to define the entire flat region by one primitive. The flat fields are regions of little or no texture and contain only one or two primitives, while well textured areas like the trailer court have hundreds of regions. Some special methods may have to be included to capture the flat areas properly.

As mentioned before several different combinations of properties were used for clustering. Some examples of these are given in the next set of figures. These and the rest of the figures to follow, show the clustered image of the primitives and the smoothed spatially generalized fully segmented image. While it is difficult to say precisely which are better, one general trend that emerged was that using 3 to 4 attributes would not necessarily give better results than using 1 to 2. As a matter of fact using four properties (properties 1, 2, 4 & 5) gave the worst result with a lot of mixing early in the processing. This was somewhat surprising as one expected that the greater the number of attributes the more is the information available for clustering. Since using more attributes increases the processing time substantially, these results suggest that one could achieve about the same quality results and save processing costs by carefully choosing a fewer number of attributes.

Figure 6.2.2 shows the clustered descending components and the spatially generalized file for properties 1, 4 and 5. This image has

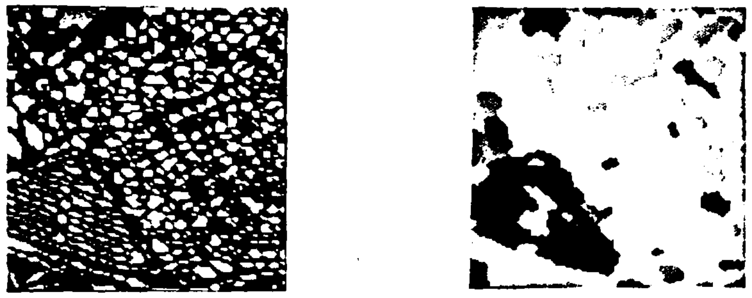


Figure 6.2.2. Descending components clustering on photo band of M4A. Properties 1, 4 and 5. Fourteen Clusters.

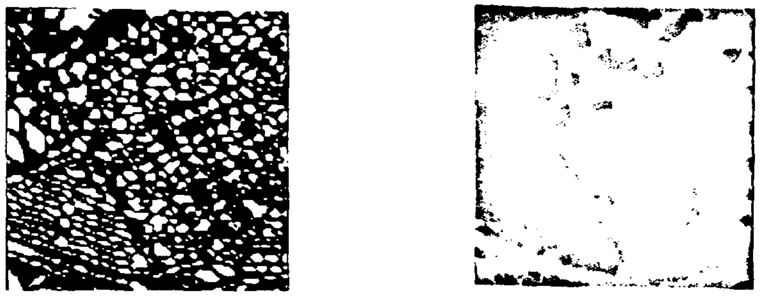
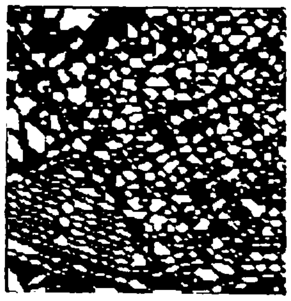


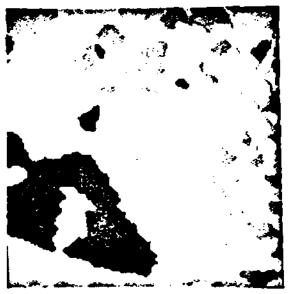
Figure 6.2.3. Descending components clustering on photo band of M4A. Property 2. Three Clusters.

fourteen groups. Out of the fourteen, eight groups contained two or less regions. Of these eight, three are large flat areas while the others can be considered noise. Thus, effectively, the results shows six textured areas. This was one of the few images in which the trailer court came out separate, but as can be seen both it and the residential area are quite broken. Further clustering only resulted in merging these two categories together. One thing to note here is that the pattern of some of the small groups in the residential area, corresponds to the light and dark shading within the same area. This pattern was also noted in a few other runs. This suggests that in these cases, because of its coarse two level texture, some of the primitives of the residential area are being treated as the flat fields; i.e. they are being clustered separately.

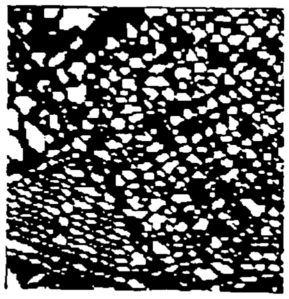
Figure 6.2.3 shows the corresponding result of using only the maximum gray tone property (property 2). The 536 primitive regions have been reduced to three groups here. No size information was used. The trailer court has merged with parts of the residential area as in Figure 6.2.1, and the bright field just above it. These are areas of similar brightness on the image. The dark fields also come out together along with some darker patches of the residential zone.

Finally figure 6.2.4 shows two iterations using the average gray tone property. These images contain seven and four clusters respectively. Of the four one is a small single region pixel. In general there is good separation between the texture classes. Unfortunately the orchard area at the bottom left has been labelled as residential, and the definition of the trailer court leaves a little


Clustered Image


Generalised Image

(a) Seven Clusters


Clustered Image

Generalised Image

(b) Four Clusters

Figure 6.2.4. Descending components clustering on photo band of M4A. Property 5. Two iterations.

more to be desired.

Next we look at the ascending components.

The results with the ascending components are poorer compared to the descending components. There are two reasons for that. First is the poor spatial definition by the ascending component primitives for the texture classes. A lot more of the pixels fell into the non-unique area and there was larger separation (black area in Figure 3.4.7b) between the primitive regions. Some areas like the top left regions hardly have any primitives. This of course would lead to inaccurate spatial generalization. The second problem resulted in poor clustering. In many runs there was considerable merging and mixing of primitives of different classes. As this was noticed quite early in the clustering processes, when the clustering thresholds are much lower, it seemed that the algorithm was having difficulty achieving good separation. The problem was that the minimum gray tones for a lot of the regions were zero. Thus they were indistinguishable on this measure. These regions were about all equally dark. For the descending components the corresponding property of maximum gray tone had a much wider variation. Some of the larger flat areas did emerge when the size property was included. However the residential part was still mixed. One item that did come out consistently was the exit ramp road, but by later stages of clustering it was usually incorporated with the residential area.

Two results for the ascending components are shown in Figures 6.2.5 and 6.2.6. The first is based on the average gray tone (property 5) and has seven clusters. Of these, two contain two or less regions. Some

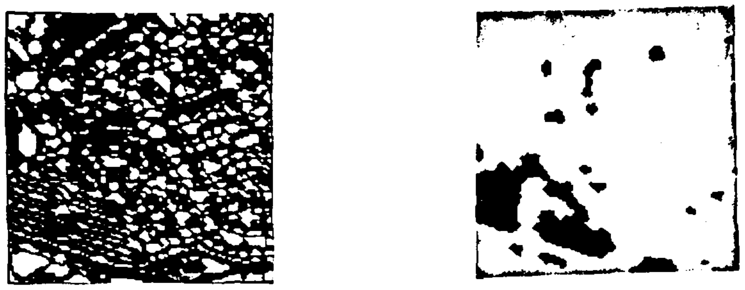


Figure 6.2.5. Ascending components clustering on photo band of M4A. Property 5. Seven Clusters.

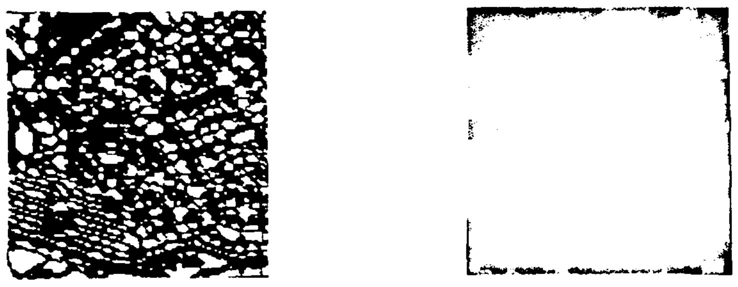


Figure 6.2.6. Ascending components clustering on photo band of M4A. Properties 2, 3 and 5. Four Clusters.

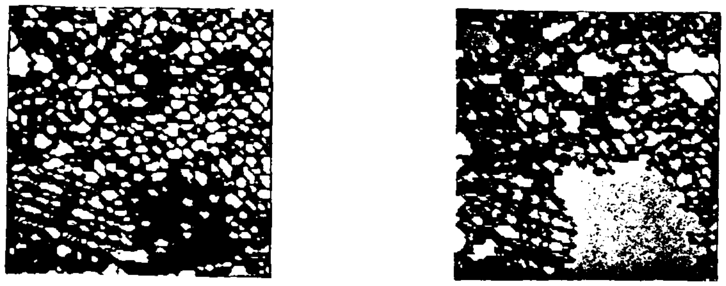
definition has been obtained for the trailer court but it is quite poor. Most of the image has been classified as residential including the fields. The second result in Figure 6.2.6 is based on properties 2, 3 and 5. There are four clusters. There is a lot of merging of the flat texture areas and again the flat fields did not separate out.

Clustering was also attempted on the radar band for image M4A. While it was consistent with the image information, the results were not as good as those from the photo image. As can be seen from Figure 3.4.1a, the radar image was quite dark in places. This fact is verified by looking at the local and true extrema of the radar band in Figure 6.2.7. The minima are red, maxima green and the transition yellow. The flat areas for the local extrema image are black. There is a large flat (with gray tone 0) area bottom right. The corresponding reachability sets are shown in Figure 6.2.8. The number of regions is smaller compared to the photo image extrema but the average size of regions is much larger. The number of descending and ascending components is 423 and 316 respectively.

The image is quite poor and as a matter of fact there is not too much texture one can determine from it. Clustering was only performed on the descending components image. The ascending components were based too much on parts of the image which had zero gray tone and did not seem too useful. Two results are shown in Figure 6.2.9 and 6.2.10. Again these images suffered from poor spatial definition for the texture classes. Also the residential area took too long to cluster causing it consistently to merge with the trailer court. Figure 6.2.9 shows the results using the size and maximum height properties (properties 1 & 2).

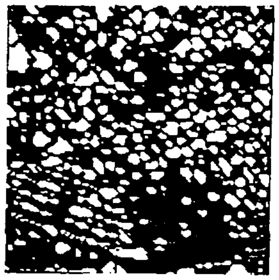
(a) Local (3x3 window) (b) True

Figure 6.2.7. Relative maxima (green) and minima (red) for the radar band of M4A.

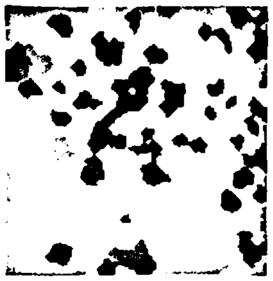


(a) Descending Components (b) Ascending Components

Figure 6.2.8. Unique reachability sets for the radar band of M4A. 423 and 316 regions respectively.

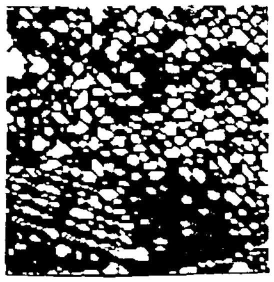


Clustered Image



Generalised Image

(a) Ten Clusters.

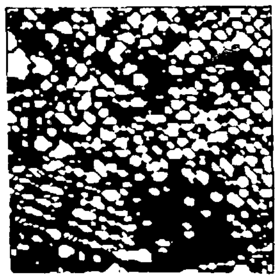


Clustered Image

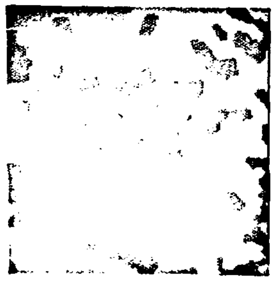
Generalised Image

(b) Three Clusters.

Figure 6.2.9. Descending components clustering on radar band of M4A. Properties 1 and 2. Two iterations.



Clustered Image



Generalised Image

Figure 6.2.10. Descending components clustering on radar band of M4A. Property 2. Three Clusters.

Two clustering iterations are shown. The first shows ten classes while the second three. The trailer court has emerged in the first image but it is still merged with parts of the residential area. Just a little more clustering yields the three class image. Even though there is a large amount of mixing, the shapes of the areas do match with the more distinct areas of the radar band.

Figure 6.2.10 is a little better result again with three classes. This is based on the maximum gray tone property. The match up between it and the high reflectance (bright) areas on the radar band is very close.

The radar band was quite disappointing for M4A. Some pre-processing attempts to enhance the radar band were also made. In one an equal probability quantization to 32 levels was done. While the image looked sharper, there was no significant change in the clustering results.

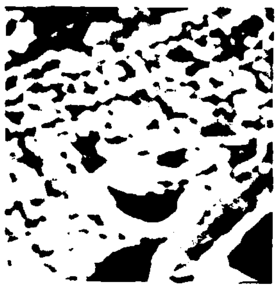
6.3 Processing Image 'M3A'

Subsection M3A extracted from the third ETL image is also 128 x 128 in size. It is over a section of the city of Fremont, Ca. and contains a residential area for the most part. The bottom right quadrant contains a creek (Alameda Creek) and there are some gravel pits in the bottom parts of the image. Across the top left corner of the picture is a railway line and a road. There are some subtle changes in texture of the residential class and it was hoped that the algorithm would pick them up.

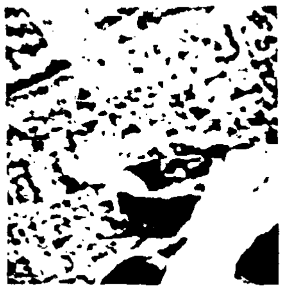
The two bands of the subsection are shown in Figure 6.3.1, with the local and true extrema in Figure 6.3.2. Again the minima are red, maxima green and the transition region yellow. The black in the local extrema image is the flat pixels. There are 483 maxima and 450 minima for the photo band of M3A. The descending and ascending components can be seen in Figure 6.3.3.

It turned out for M3A the ascending components gave better results over the descending components. A major reason for this was the better spatial definition of the different areas of the image by the ascending components. The gravel pits which are dark are almost completely missed by the descending components, but were picked up fairly well by the ascending components. One of them just below the center was not picked up by either set as it did not contain an extremum of either kind. The pits gave rise to the same problems encountered for the fields of M4A. Again we are using one or two primitives to define a 'texture' class. The problem is exaggerated as the size of these regions is larger and there are more of them. Furthermore the clustering ran into the problem here that even though the pits are all the same class, their sizes differed tremendously. Thus when the size attribute was used, the regions usually remained separate. Grouping only took place when the size was not included. This is an example of an attribute acting detrimentally. Size and shape properties for these large regions are also affected by picture edges and thus would be questionable to use. For example the pit at the bottom right is cut by the picture edge and has an artifical shape.

The creek also presented similar problems, though it is easily apparent to the eye. Again it has very little texture and very few extrema that define it. An examination of Figure 6.3.3 shows that we

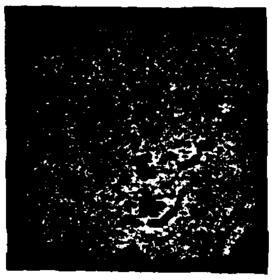


(a) Radar band



(b) Photo band

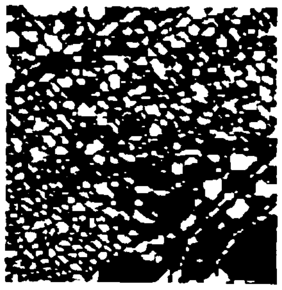
Figure 6.3.1. Image M3A. Size 128x128.



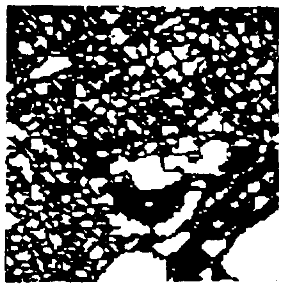
(a) Local (3x3 window)

(b) True

Figure 6.3.2. Relative maxima (green) and minima (red) for the photo band of M3A.



(a) Descending Components



(b) Ascending Components

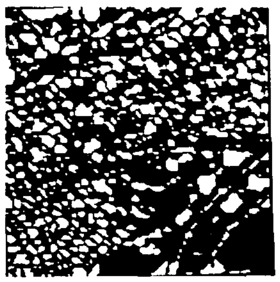
Figure 6.3.3. Unique reachability sets for the photo band of M3A. 483 and 450 regions respectively.

pick up more of the bank than we do of the river, owing to its brighter definition.

Processing with the size and maximum height properties on the descending components, which were the best for M4A, is shown in Figure 6.3.4. This is quite poor as there is very little definition for the major regions. The result is the uncontrolled growth of some of the clusters into the large unclassified area of the gravel pits and the river regions. The image contains eight clusters. A better picture is shown in Figure 6.3.5. Based on the average gray tone property this image has four clusters. There is still not a good definition of the pits but some distinction has been achieved for the slightly differing areas of the residential part. The school area which is the bright spot just left of the center of the image was kept quite separate from the rest of the residential tract, but unfortunately was confused with the bright levels of the river region.

Two examples of processing with the ascending components are shown in the next two figures. Figure 6.3.6 shows seven groups based on the number of levels and the average gray tone properties. The definition of the pits and the river is much better. However merging has taken place with parts of the residential zone. Figure 6.3.7 shows the results of using three properties: maximum level, minimum level and the average level. It is similar to the previous one but was unable to get all the pits together. There are eight classes on this image.

Processing on the radar band was also performed for this image. For comparison Figures 6.3.8 and 6.3.9 show the local and true extrema and the corresponding reachability sets. As before there are fewer


Clustered Image

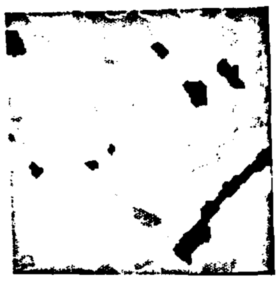
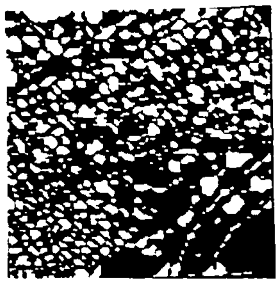
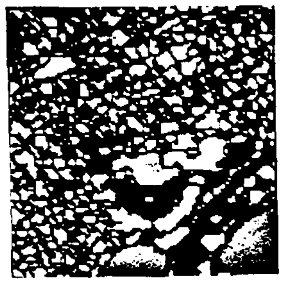

Generalised Image

Figure 6.3.4. Descending components clustering on photo band of M3A. Properties 1 and 2. Eight Clusters.

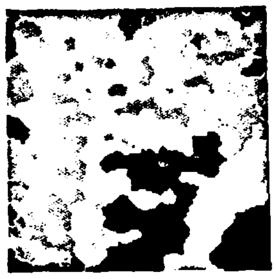

Clustered Image

Generalised Image

Figure 6.3.5. Descending components clustering on photo band of M3A. Property 5. Four Clusters.

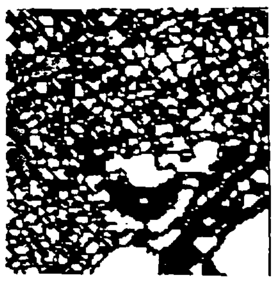


Clustered Image

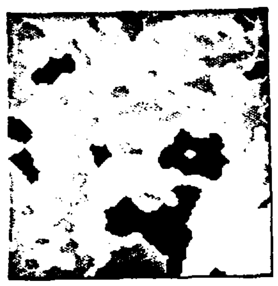


Generalised Image

Figure 6.3.6. Ascending components clustering on photo band of M3A. Properties 4 and 5. Seven Clusters.



Clustered Image



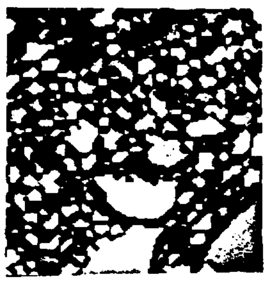
Generalised Image

Figure 6.3.7. Ascending components clustering on photo band of M3A. Properties 2, 3 and 5. Eight Clusters.

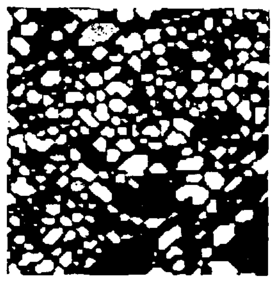
(a) Local (3x3 window)

(b) True

Figure 6.3.8. Relative maxima (green) and minima (red) for the radar band of M3A.

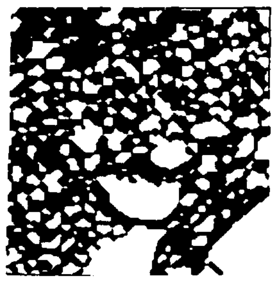


(a) Descending Components



(b) Ascending Components

Figure 6.3.9. Unique reachability sets for the radar band of M3A. 274 and 226 regions respectively.



Clustered Image

Generalised Image

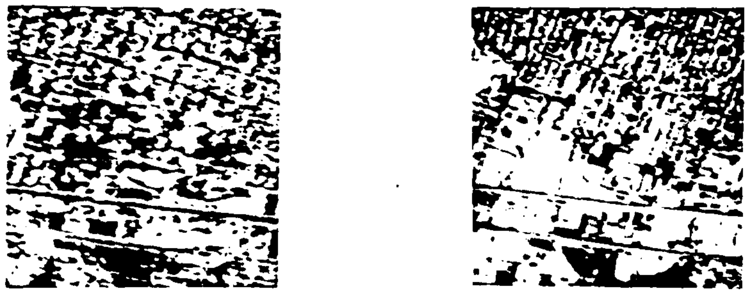
Figure 6.3.10. Ascending components clustering on radar band of M3A. Property 2. Four Clusters.

number than those from the photo band and the sets are larger in size. The image contains 274 descending and 226 ascending components. The processing results were not very impressive. One example is shown in Figure 6.3.10. It is based on the maximum gray level on the ascending components image. It has four classes. Only a few major areas can be identified.

6.4 Processing Image 'M6A'

M6A is a subsection of the sixth ETL image. This image is over a section of Oakland, Ca. and is 200 x 200 pixels in size. The image is shown in Figure 6.4.1. Owing to its larger size, it and all of its processed results were viewed at half the scale (on the TV monitor), compared to images M4A and M3A. The image was chosen to see how well the algorithm could distinguish between different man-made textures. The picture contains some houses in city blocks at the top. These show as fine texture in the top area and other parts of the image. Two railway lines run across the bottom part and are surrounded by large buildings.

Figure 6.4.2 shows the local and true extrema. In Figure 6.4.3 we have the reachability sets. There are 1152 maxima and 1146 minima regions for the photo band. An interesting fact was noted in the local extrema image of Figure 6.4.2. In the original image the city blocks are defined by streets which run at an angle of about 23 degrees east of the vertical. In the local extrema image this inclination is not apparent. Instead there is a slight pattern of straight lines about 6 degrees west of the vertical! The pattern disappears in the true extrema image with the pattern of the original inclination reappearing

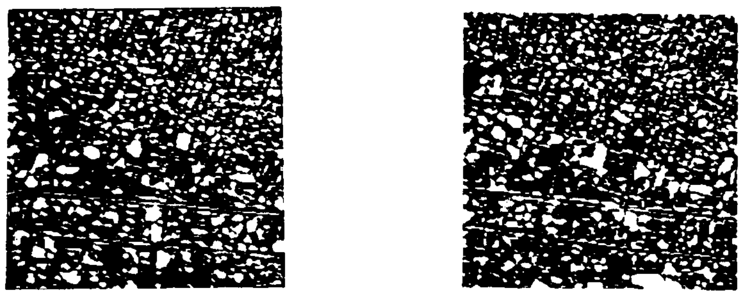


(a) Radar band (b) Photo band

Figure 6.4.1. Image M6A. Size 200x200.

(a) Local (3x3 window) (b) True

Figure 6.4.2. Relative maxima (green) and minima (red) for the photo band of M6A.



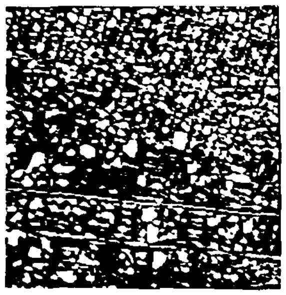
(a) Descending Components (b) Ascending Components

Figure 6.4.3. Unique reachability sets for the photo band of M6A. 1152 and 1146 regions respectively.

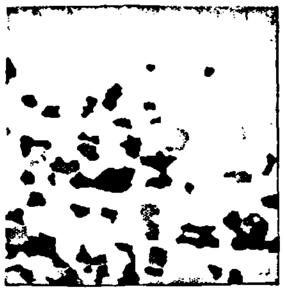
slightly. The original pattern is readily apparent in the reachability set images. There was no clear explanation for this.

Some examples of processing with the descending and ascending components are given in the next set of figures. Figure 6.4.4 shows the results of processing attributes 1 and 2, the size and maximum height. This is on the descending components image. There are thirteen clusters shown on this. Of these seven contain eight or fewer regions out of the original 1152 regions. Figure 6.4.5 shows the clustering process which used only the maximum height property. The picture has five classes. In both of these the city blocks come out pretty well as they constitute a well defined texture. Few individual buildings were also distinguishable. The railway lines which were so well defined on the reachability sets were however always lost. They merged with the residential area no matter which properties were being used. These and other linear features are hard to extract, as it is hard to define texture on them. Using shape and orientation features could perhaps keep them separate.

In Figures 6.4.6 and 6.4.7 we have two runs on processing the ascending components. The first was on four properties: size, maximum height, number of levels and the average gray tone. It has sixteen classes of which ten consisted of four or fewer regions. The railway lines were again absorbed by the residential class, but the buildings between them were quite distinct. Figure 6.4.7 shows a clustering example with ten classes, based on the maximum, minimum and the average level properties. Five of these are small clusters. The residential area is still two sections which correspond in part to the finer and

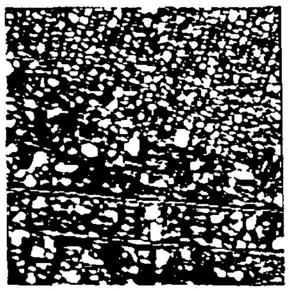


Clustered Image



Generalised Image

Figure 6.4.4. Descending components clustering on photo band of M6A. Properties 1 and 2. Thirteen Clusters.



Clustered Image

Generalised Image

Figure 6.4.5. Descending components clustering on photo band of M6A. Property 2. Five Clusters.

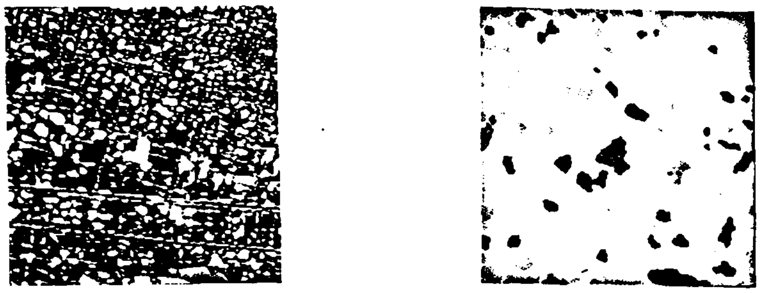


Figure 6.4.6. Ascending components clustering on photo band of M6A. Properties 1, 2, 4 and 5. Sixteen Clusters.

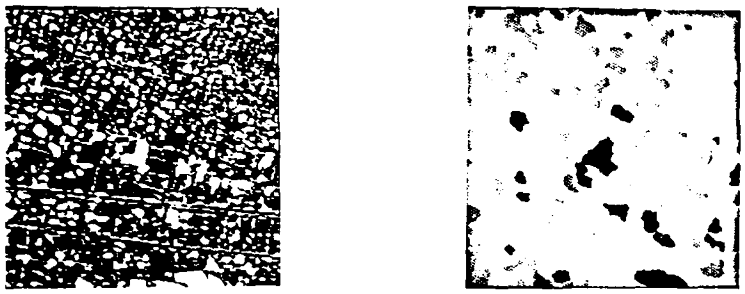


Figure 6.4.7. Ascending components clustering on photo band of M6A. Properties 2, 3 and 5. Ten Clusters.

coarser textures of that domain.

No processing was done with radar band for this image.

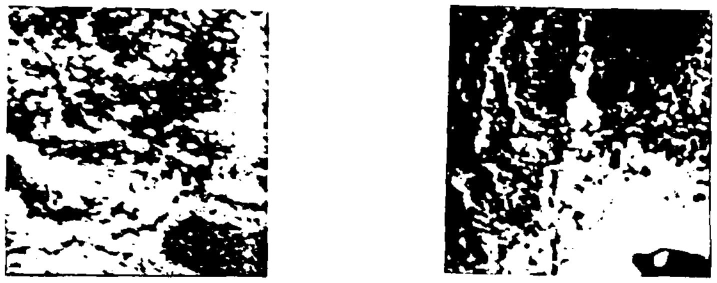
6.5 Processing Image 'M1A'

The fourth image presented here is from the first ETL image. It is again 128 x 128 pixels in size. The image is located at the southern tip of Crystal Springs reservoir, west of San Carlos, Ca. It was chosen for its wooded hillside or forest textures which cover most of the top and left parts of the image. In the bottom right quadrant there is a beach and a little bit of the lake. Between the beach and the hillside is another swath of land refered to here as the upper beach. It contains a lighter density of trees and is a low lying area of the hillside.

Both the radar and photo bands were processed. Owing to the different acquisition dates the radar image shows a larger water area than the photo image. This can be seen in Figure 6.5.1 which shows both these bands. Figure 6.5.2 shows both the local and true extrema for the photo band, while Figure 6.5.3 shows the reachability sets. There are 670 descending and 593 ascending components on this image.

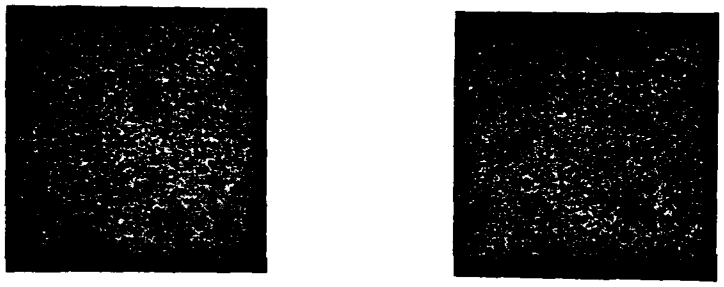
The image is quite difficult to analyse as the changes in the texture of the forest if any are quite subtle. Also parts of the photo image are quite dark making it hard to see certain regions. The algorithm performed quite well however. The radar band on this image is quite bright, a fact which is reflected in the processing of that data.

On the photo band the ascending components clustering was not too good and the best result obtained is shown in Figure 6.5.4. This image



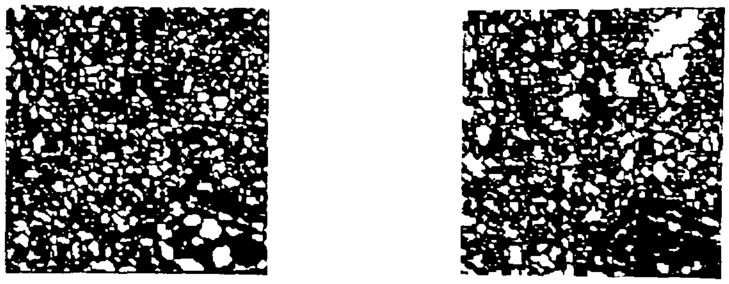
(a) Radar band (b) Photo band

Figure 6.5.1. Image M1A. Size 128x128.



(a) Local (3x3 window) (b) True

Figure 6.5.2. Relative maxima (green) and minima (red) for the photo band of M1A.



(a) Descending Components (b) Ascending Components

Figure 6.5.3. Unique reachability sets for the photo band of M1A. 670 and 593 regions respectively.

has seven classes. The clustering was done using the size and maximum height properties. Most of the area came out as one class. The sea was merged with the forest as they are both equally dark. Only the immediate beach came out as one separate class. The upper beach is patchy. The few isolated regions in the forest correspond to the large size ascending components regions (see Figure 6.5.3). They could not be clustered with the many smaller sized regions of the forest area.

The descending components clustering was quite good except for the sea which came out in two parts. Two iterations are shown in Figure 6.5.5. These are based on the maximum height attribute. In Figure 6.5.5a there are six classes. These reflect quite closely the different regions of the image. The forest area is in two parts. The light blue on the green correspond to the bright ridges that can be seen on the photo band. The upper beach came out quite well, while the main beach is in two segments. Only the sea was poor. As may be seen from the descending components image (Figure 6.5.3), there are very few primitives defining the dark flat area. This is of course the same problem encountered in previous images.

In the next iteration, Figure 6.5.5b, the forest has been reduced to one category. This image contains four classes. The sea is still broken and now partly merged with the forest as they are both quite dark. Overall though it is a good match up on shapes of the classes.

While the radar results are not as pleasing as those from the photo band, the algorithm again performed well. As mentioned before the radar image was taken at a different date and shows more water. The beach and

Clustered Image
Generalised Image

Figure 6.5.4. Ascending components clustering on photo band of M1A. Properties 1 and 2. Seven Clusters.

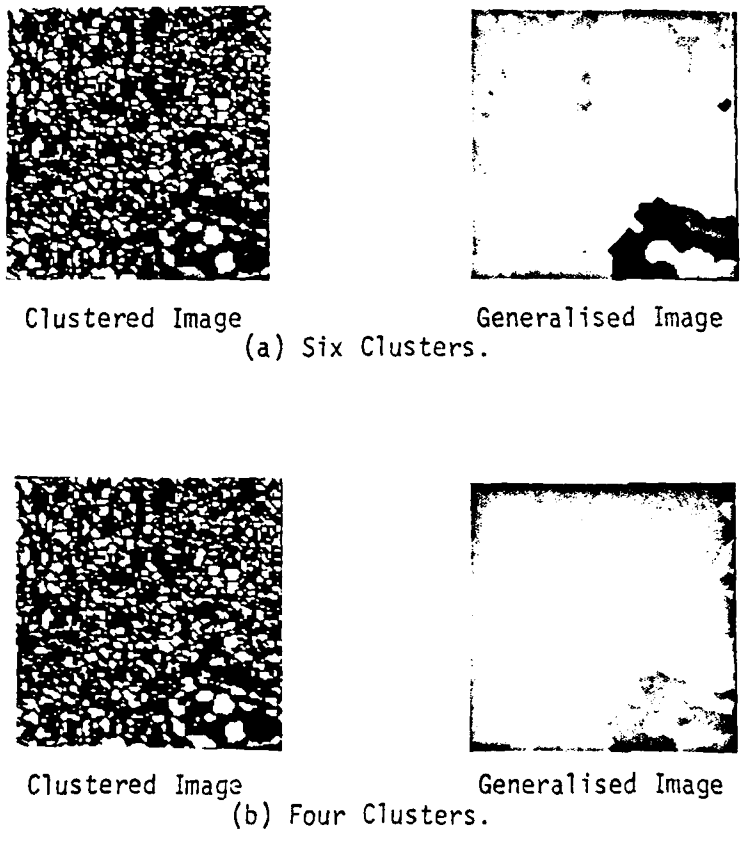


Figure 6.5.5. Descending components clustering on photo band of M1A. Property 2. Two iterations.

the upper beach areas are not distinct at all. Furthermore it is quite noisy. This actually was good as it resulted in many more extrema than the photo band, enabling us, among other things, to get a better definition of the sea area. Figure 6.5.6 shows these and in Figure 6.5.7 we have the corresponding reachability sets. There are 813 maxima and 881 minima in this band.

In Figure 6.5.8 we have the clustering based on the size and the maximum height property for the descending components of the radar band. This image has five groups. It is hard to separate different texture regions on this band. The different colors here show more the different brightness levels as there is not too much variation in the sizes. The match is fairly good. The light blue and green correspond to the brighter areas, while the purple and pink correspond to the much darker regions. The sea here has more body, but is still merged with the dark areas of the forest.

Figure 6.5.9 show another clustering session on the descending components. This is based on the minimum gray level attribute. There are four classes in this image. It turned out a little better. Again the green and light blue are the brighter tracts and the purple the darker ones on the radar image.

Finally in Figure 6.5.10 we have a result on processing the ascending components for the radar data, based on the maximum height property. This image also has four classes. It is not as good as the previous one as the definition of the sea is bad. However the shading patterns of the forest are still represented faithfully.

(a) Local (3x3 window) (b) True

Figure 6.5.6. Relative maxima (green) and minima (red) for the radar band of M1A.

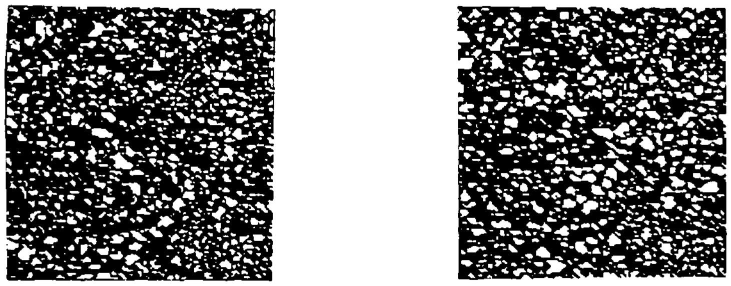

(a) Descending Components (b) Ascending Components

Figure 6.5.7. Unique reachability sets for the radar band of M1A. 813 and 881 regions respectively.

Clustered Image
Generalised Image

Figure 6.5.8. Descending components clustering on radar band of M1A. Properties 1 and 2. Five Clusters.

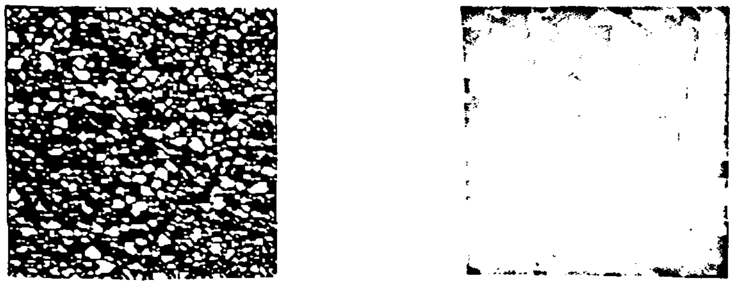


Figure 6.5.9. Descending components clustering on radar band of M1A. Property 3. Four Clusters.

Clustered Image
Generalised Image

Figure 6.5.10. Ascending components clustering on radar band of M1A. Property 2. Four Clusters.

7.0 CONCLUSION AND RECOMMENDATIONS

In the last few chapters we looked at a few techniques which used image extrema and the reachability set primitives for quantifying texture. Our goal was to examine the properties of these primitives and develop schemes to segment images based only on texture. To this end the experiments of Chapters 4 and 6 were quite successful, although not without problems. In this final section we will briefly review some of these problems and at the same time look at suggestions for further work.

Segmentation based on the density images of Chapter 4 is not very evident and it wasn't expected to be. The extrema density is a weak texture measure and is unlikely it will give rise to good direct segmentation. It does however separate regions of fine and coarse texture quite well and has been successfully used in texture discrimination experiments (Mitchel et al 1977, 1978). The main aim here was to see how well it suited up to other measures such edge density, and was found to be generally comparable. Its main segmentation application lies mainly as a texture discriminator band in a multispectral clustering process. Its advantage lies in the fact that it is fast and straight forward to generate.

The properties of the primitives turned out to be more useful for segmentation. The overall results of clustering based on the property values of the primitives were quite good. The examples of Chapter 6 show the procedure worked very well in separating regions of different texture, but had difficulty with the flat homogeneous areas of the image. As mentioned before this is partly due to the fact that we are trying to define the spatial domain of the flat region by one or two primitives. The reachability set primitives were not intended for this, as it is not really meaningful in the context in which we are using them. They are supposed to represent a primitive of the texture component and not the region itself. Using it for the latter purpose gives rise to problems. Not only do we have different spatial definition on the dark and light fields using ascending and descending components, but some of the properties (size, shape) are no longer comparable with the texture region primitives. All this of course leads to errors in clustering and the spatial generalization process.

The inability of the process to handle the flat regions adequately does not reflect badly on it when we remember that the aim was to use the primitives to define components of texture. The uniform gray tone areas are really regions of no texture and do not fall into the main class of items we wanted to study.

The flat region problem is not unsolvable. The most direct method would be to extract the low texture homogenous tone areas by spatial clustering. There are several ways to do that, Singh (1977) being one such scheme. Once this is done, the texture image can be segmented without the homogenous regions. The latter may then be merged with the segmented picture, either as is or after clustering has been performed on them. This clustering would be on the tonal properties of the flat regions. The merging would yield the final classified image.

Another way to tackle this problem is by the use of noise. The trouble with the flat areas is that there are very few local extrema and the few that there are result in large ungainly reachability sets. By adding a little noise to these regions, we seed the flat area with artificial extrema. This not only increases the number of extrema but also breaks up the original large reachability sets into smaller ones. The size of these would be a function of the spatial distribution of the noise pixels. This approach is very well illustrated for the sea region for the photo and radar bands of M1A (Figures 6.5). The radar band is quite noisy compared to the photo band and the resulting extrema break up the sea into many small reachability sets. This gives a better spatial definition for the sea which was lost for the smoother photo band.

A more complex problem is the one of two level textures. This refers to regions which contain two types of textures or in which the textured is two-layered. An example of this would be fine micro texture areas arranged in some manner to form a coarse macro texture region. Another example would be a texture component which consists of an arrangement of a set of three or more light and dark shapes. These sets are arranged over a region to give it its texture. The reachability sets may capture each of these shapes separately when they actually should be treated together. During the clustering the different subshapes of each set will cluster with the corresponding subshapes from other sets. The segmented image will then show the texture region broken according to the pattern of these subshapes, rather than as one region as we would like. The problem involves defining not only the dimensionality of the texture but also size of the basic texture component. It is partly one of choosing the correct resolution level. To some degree it can be solved by increasing the thresholds in the clustering, but there is no guarantee in that. At some stage either at the beginning or on the clustered-segmented image, spatial and global techniques will have to be applied.

This problem of two level textures was not tackled here and is one area for further research. About the only example encountered was the residential area of the M4A. From a broad point of view it represents a coarse texture area. However on closer examination it shows a pattern of light and dark shapes. It was noted during most of the processing runs, that this entire region broke into three or four classes and only at a very late stage in the clustering did it reduce to a smaller number. The three of four classes pretty much followed the arrangement of the shapes, giving the segmented image a patchy appearance.

There are two other points which need further mentioning. One is the use of the spatial generalization process as this is a potential source of error. As discussed in Section 6.1 classification errors may result if in an area the number of extrema are few and the distances between them large. The spatial generalization could then generate an artificial growth pattern giving rise to errors in classification. One such type of area we have already encountered are the flat no texture regions of the image. However on closer examination, other areas with slowly varying graytones also exhibit a similar property. They have few extrema placed far apart, with the result that a lot of pixels fall into the non-unique reachability class. These are all regions with low texture some of them being uniform slopes. A more controlled growth process is then needed for such regions, one that is guided by the sloped facets and flats of the image. This would however slow the growing operation.

The second deals with the number of attributes used in the clustering. As pointed out earlier, one of the facts that emerged was that using three or four attributes did not necessarily improve the result over using one or two. The choice of the attributes here is of course important but the result is still surprising. The original hypothesis was that by increasing the number of attributes, we should do no worse. However this did not hold out. The reasons for it are not clear and need to be established. This is important if we want to optimise the clustering performance. Balance has to be achieved between choosing enough attributes to be able to discriminate between categories, and at the same time keeping this number small for computational efficiency.

A natural extension for using the extrema primitives has been described in Section 5.4. This would be based on the work of Davis et al (1979) and uses the extrema in the framework of a strong texture measure. Davis et al used local maxima edge points with directional properties to generate Generalised Co-occurence Matrices (GCMs). These were shown to be more powerful than the regular gray level co-occurence matrices for texture discrimination. It is felt that GCMs based on the extrema primitives with the attributes of Chapter 5, should work just as well if not better.

The results of this project show that the extrema and their derivatives are useful tools for texture analysis. They were used successfully for both image discrimination as well as segmentation. Further work however is necessary to reduce errors in classification and eliminate the problems encountered.

REFERENCES

Anderson, M., "Cluster Analysis for Applications," Academic Press, New York, 1973.

Bachi, Roberto, "Geostatistical Analysis of Territories," IEEE Transactions on System, Man, anu Cybernetics, Vol. SMC-8, No. 2, February 1978.

Bavel, Z., "Introduction to the Theory of Automata and Sequential Machines," 1968, to be published.

Bryant, W., "KANDIDATS-II Kansas Digital Image Data System," RSL Technical Report 0920-2, CRINC, University of Kansas, September 1976.

Chen, P. and T. Pavlidis, "Segmentation by Texture Using a Co-occurrence Matrix and a Split-Merge Algorithm," Technical Report 237, Princeton University, Princeton, New Jersey, January 1978.

Chow, C. and T. Kaneko, "Boundary Detection of Radiographic Images by Threshold Method," Frontiers of Pattern Recognition, Academic Press, 1972.

Davis L., S. Johns, and J.K. Aggarwal, "Texture Analysis Using Generalized Co-occurrence Matrices," Pattern Analysis and Machine Intelligence, Vol. PAMI-1, No. 3, July 1979.

Ehrich, R. and J.P. Foith, "Representation of Random Waveforms by Relational Trees," IEEE Transactions on Computers, Vol. C-25, No. 7, July 1976.

Ehrich, R. and J.P. Foith, "A View of Texture Topology and Texture Description," Computer Graphics and Image Processing, Vol. 8, 1978.

Frolov, Y.S., "Measuring the Shape of Geographical Phenomena: A History of the Issue," Soviet Geography: Review and Translation, Vol. XVI, No. 10, December 1975.

Galloway, M., "Texture Analysis Using Gray Level Run Lengths," Computer Graphics and Image Processing, Vol. 4, 1974.

Haralick, R.M., "A Texture-Context Free Extraction Algorithm for Remotely Sensed Images," Proceedings 1974 IEEE Decision and Control Conference, Gainsville, Florida, 15-17 December 1971.

Haralick, R.M., "A Textural Transform for Images," Proceedings of the IEEE Conference on Computer Graphics, Pattern Recognition, and Data Structure, Beverly Hills, California, 14-16 May 1975.

Haralick, R.M., "Statistical and Structural Approaches to Texture," Proceedings of the IEEE, Vol. 67, No. 5, May 1979.

Haralick, R.M. and L.G. Shapiro, "The Consistent Labeling Problem," Pattern Analysis and Machine Intelligence, Vol. PAMI-1, No. 2, April 1979.

Haralick, R.M., K.S. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-3, November 1973.

Hartigan, J.A., "Clustering Algorithms," Wiley, New York, 1975.

Johnson, D., "KANDIDATS - An Interactive Image Processing System," Masters Thesis, Electrical Engineering Department, University of Kansas, 1974.

Lu, S.Y. and K.S. Fu, "A Syntactic Approach to Texture Analysis," Computer Graphics and Image Processing, Vol. 7, No. 3, July 1978.

McCormick, B.H. and S.N. Jayaramamurthy, "Time Series Model for Texture Synthesis," International Journal of Computer and Information Sciences, Vol. 3, No. 4, December 1974.

Mitchell, O.R. and S.G. Carlton, "Image Segmentation Using a Local Extrema Texture Measure," Pattern Recognition, Vol. 10, 1978.

Mitchell, O., C. Myers, and W. Boyne, "A Max-Min Measure for Image Texture Analysis," IEEE Transactions on Computers, Vol. C-25, No. 4, April 1977.

Peucker, T. and D. Douglas, "Detection of Surface Specific Points by Local Parallel Processing of Discrete Terrain Elevation Data," Computer Graphics and Image Processing, Vol. 4, No. 4, December 1975.

Pickett, R.M., "Visual Analyses of Texture in the Detection and Recognition of Objects," Picture Processing and Psychopictorics, Edited by Lipkin and Rosenfeld, Academic Press, New York, 1970.

Read, J.S. and S.N. Jayaramamurthy, "Automatic Detection of Texture Feature Detectors," IEEE Transactions on Computers, Vol. C-21, No. 7, July 1972.

Rosenfeld, A. and M. Thurston, "Edge and Curve Detection in Visual Scene Analysis," IEEE Transactions on Computers, Vol. C-20, No. 5, May 1971.

Rosenfeld, A. and E. Troy, "Visual Texture Analysis," Technical Report 70-116, University of Maryland, July 1970.

Serra, J. and G. Verchery, "Mathematical Morphology Applied to Fibre Composite Materials," Film Science and Technology, Vol. 6, 1973.

Singh, A., "KANDIDATS/CLUSTER Non-Supervised Digital Image Clustering Using Spatial Information," Masters Thesis, Computer Science, University of Kansas, February 1977.

Sneath, R.R. and P.A. Sokal, "Principles of Numerical Taxonomy," W.H. Freeman and Company, 1963.

Toriwaki, J. and T. Fukumura, "Extraction of Structural Information from Grey Pictures," Computer Graphics and Image Processing, Vol. 7, No. 1, 1978.

APPENDIX A

COLOR TABLE FOR CLUSTERED IMAGES OF CHAPTER 6.

The list below gives the colors used to show the different classes of the clustered images of Chapter 6. There are twenty colors for upto twenty cluster codes. If an image has more than twenty classes, cluster codes twenty and higher have the same color.

1 -	Light Green	11 -	Red
2 -	Light Blue	12 -	Ochre
3 -	Purple	13 -	Yellow Green
4 -	Pink	14 -	Dark Green
5 -	Orange	15 -	Blue Green
6 -	Yellow	16 -	Dark Purple
7 -	White	17 -	Dark Red
8 -	Green Blue	18 -	Dark Orange
9 -	Dark Blue	19 -	Dark Yellow Green
10 -	Light Purple	20 -	Dark Gray

APPENDIX B

IMAGES USED FOR CLUSTERING EXPERIMENTS.

Image:	M4A
Source:	ETL files 7 & 8
Image Size:	128 x 128 pixels
Subsection Coordinates from ETL Image	
First & Last Rows:	257 to 384
First & Last Cols:	321 to 448
Acquisition Dates	
Photo Band:	21-Oct-71
Radar Band:	16-Mar-71
Approximate Geographic: Location	Section of Union City, North of Newark, Ca.

Image histograms on following pages.

```
1----------------------------------------------------------1
1                                                          1
1                    HISTOGRAM OF                          1
1                                                          1
1    M4ARSXS44  BAND     1    RADAR BAND.                  1
1                                                          1
1             INTERVAL SIZE 0.254E+01                      1
1             DATA STARTS AT    0  AND ENDS AT  127        1
1             NUMBER OF DATA POINTS     16384              1
1             MEAN= 0.279E+02     VARIANCE= 0.952E+03      1
1                                                          1
1 PROBABILITY                                              1
1  0.300 I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1  0.249 I                                                 1
1        I                                                 1
1        I                                                 1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1  0.197 I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1  0.146 I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1  0.094 I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I** *                                             1
1        I** *                                             1
1        I**** *                                           1
1        IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
1             I         I         I         I         I    1
1             13       38        64        89       114    1
1                                                          1
1                        DATA VALUES                       1
1                                                          1
1----------------------------------------------------------1
```

```
                    HISTOGRAM OF

    M4ARSXS44  BAND    2     PHOTO BAND.

              INTERVAL SIZE 0.170E+01
              DATA STARTS AT    0  AND ENDS AT   85
              NUMBER OF DATA POINTS      16384
              MEAN= 0.411E+02      VARIANCE= 0.223E+03
```

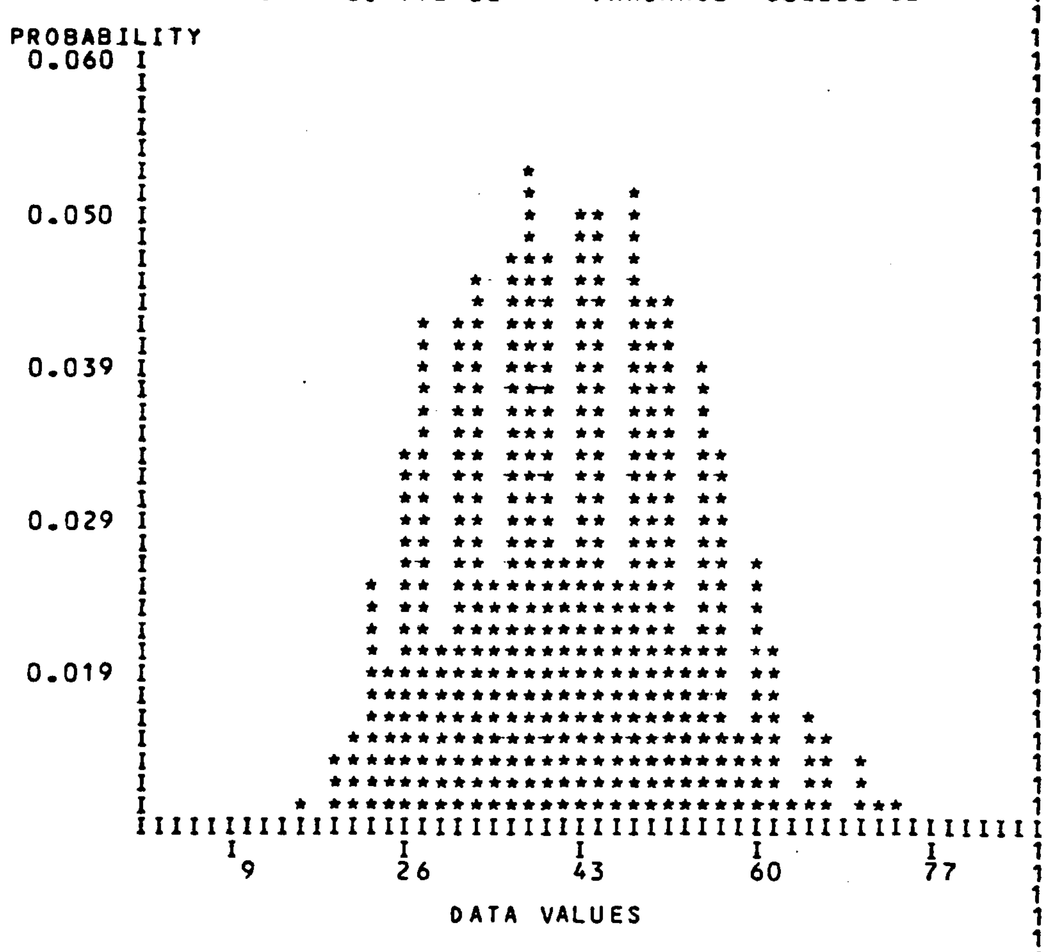


Image:	M3A
Source:	ETL files 5 & 6
Image Size:	128 x 128 pixels
Subsection Coordinates from ETL Image	
First & Last Rows:	65 to 192
First & Last Cols:	65 to 192
Acquisition Dates	
Photo Band:	21-Oct-71
Radar Band:	16-Mar-71
Approximate Geographic: Location	Section of Freemont, Ca.

Image histograms on following pages.

```
1----------------------------------------------------------1
1                                                          1
1                    HISTOGRAM OF                          1
1                                                          1
1     M3ARSXS44 BAND     1    RADAR BAND.                  1
1                                                          1
1              INTERVAL SIZE 0.232E+01                     1
1              DATA STARTS AT    0  AND ENDS AT  116       1
1              NUMBER OF DATA POINTS    16384              1
1              MEAN= 0.293E+02    VARIANCE= 0.430E+03      1
1                                                          1
1 PROBABILITY                                              1
1  0.200 I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1  0.166 I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1  0.131 I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I                                                 1
1        I*                                                1
1  0.097 I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1        I*                                                1
1  0.063 I*     *  *                                       1
1        I*     *  *                                       1
1        I*     *  *  *                                    1
1        I*     *  *  *  *                                 1
1        I*  *  ** *  *  *                                 1
1        I*  *  **********  *                              1
1        I*  *************  *                              1
1        IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
1             I         I         I         I         I    1
1             12        35        58        81       104   1
1                                                          1
1                         DATA VALUES                      1
1                                                          1
1----------------------------------------------------------1
```

```
                                 HISTOGRAM OF

      M3ARSXS44  BAND     2      PHOTO BAND.

                     INTERVAL SIZE 0.210E+01
                     DATA STARTS AT     0  AND ENDS AT   105
                     NUMBER OF DATA POINTS      16384
                     MEAN= 0.373E+02     VARIANCE= 0.470E+03

 PROBABILITY
  0.100 I
        I
        I*
        I*
        I*
        I*
        I*
  0.083 I*
        I*
        I*
        I*
        I*
        I*
        I*
  0.066 I*
        I*
        I*
        I*
        I*                   *
        I*                   *
        I*                   *
  0.049 I*                   *
        I*                   *
        I*                   *
        I*                   *
        I*                   ****
        I*         *      * *******
        I*         *    ***********
  0.031 I*         *    *************
        I*         *    *************
        I*         *  * **************
        I*         ******************* *
        I* *       *********************
        I* *      **********************
        I* *    ************************
        IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
             I         I         I         I         I
             11        32        53        74        95

                         DATA VALUES
```

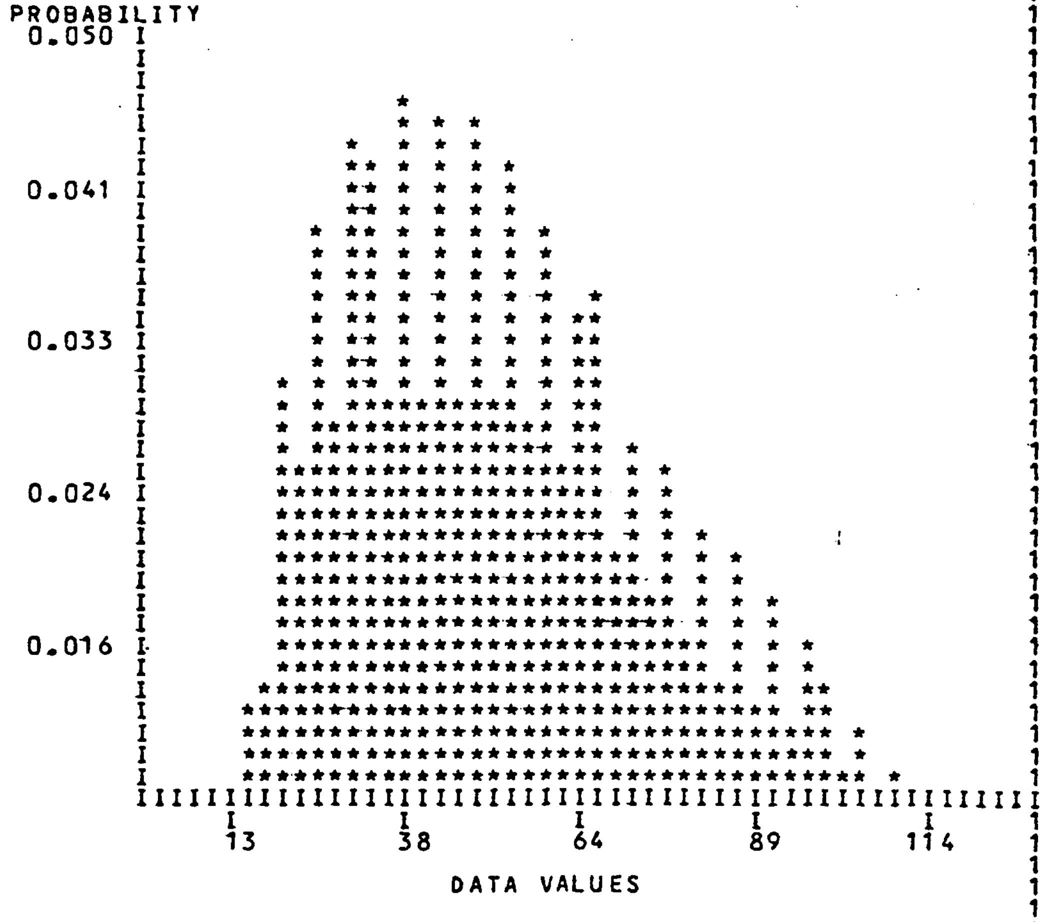
Image:	M6A
Source:	ETL files 11 & 12
Image Size:	200 x 200 pixels
Subsection Coordinates from ETL Image	
First & Last Rows:	1 to 200
First & Last Cols:	313 to 512
Acquisition Dates	
Photo Band:	21-Oct-71
Radar Band:	16-Mar-71
Approximate Geographic: Location	Section of Oakland, Ca.

Image histograms on following pages.

```
                    HISTOGRAM OF

   M6ARSXS44  BAND     1      RADAR BAND.

              INTERVAL SIZE 0.254E+01
              DATA STARTS AT     0  AND ENDS AT  127
              NUMBER OF DATA POINTS      40000
              MEAN= 0.544E+02     VARIANCE= 0.678E+03
```



```
                          HISTOGRAM OF

     M6ARSXS44  BAND    2     PHOTO BAND.

                 INTERVAL SIZE 0.208E+01
                 DATA STARTS AT     0  AND ENDS AT  104
                 NUMBER OF DATA POINTS      40000
                 MEAN= 0.385E+02     VARIANCE= 0.339E+03

 PROBABILITY
 0.060 I
       I
       I
       I            *
       I            *
       I            *
       I            *
 0.050 I            *
       I            *
       I            *   ****
       I            ** ******
       I            *********    *
       I            ***********  *
       I            ***********  *
 0.039 I            ***********  *
       I            ***********  *
       I            ***********  *
       I            ***********  *
       I            ************ *
       I            ************ *
       I           ***************
 0.029 I*         ****************
       I*         ****************
       I*        *****************
       I*        *****************
       I*        ******************
       I*       *******************
       I*       ********************
 0.019 I*      **********************
       I*      **********************
       I*      ***********************
       I*     *************************
       I*     **************************
       I*   ****************************
       I*   *****************************
       IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
            I         I         I         I         I
            10        31        52        73        94

                          DATA VALUES
```

Image:	M1A
Source:	ETL files 1 & 2
Image Size:	128 x 128 pixels
Subsection Coordinates from ETL Image	
First & Last Rows:	129 to 256
First & Last Cols:	385 to 512
Acquisition Dates	
Photo Band:	25-Nov-75
Radar Band:	15-Oct-74
Approximate Geographic: Location	Southern tip of Crystal Springs resevoir, West of San Carlos, Ca.

Image histograms on following pages.

```
                         HISTOGRAM OF

     M1ARSXS44  BAND     1      RADAR BAND.

               INTERVAL SIZE 0.252E+01
               DATA STARTS AT     1  AND ENDS AT  127
               NUMBER OF DATA POINTS      16384
               MEAN= 0.460E+02     VARIANCE= 0.502E+03

 PROBABILITY
  0.060 I         *
        I         *
        I         *
        I         *
        I         * *
        I       * * *
        I       * * *
  0.050 I       * * *
        I       * * *
        I       * * * *
        I       * * * *
        I       * * * * *
        I       *** * * * *
        I       *** * * * *
  0.039 I       ***** * * * *
        I       ***** * * * *
        I       ***** * * * * *
        I       ******* * * * *   *
        I       ******* * * * *   *
        I       ******* * * * * * *
        I       *********** * * * **
  0.029 I     * *********** *** * ** *
        I     * *********** *** * ** *
        I     ***************** * ** *
        I     ***************** * ** *
        I     ******************* ** *
        I     ********************** *
        I     ********************** * *
  0.019 I     ************************ *
        I     ************************ *
        I     **************************
        I     ************************** *
        I     ****************************
        I    ***************************** *
        I    ******************************* *
        IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII
             I         I         I         I         I
             14        39        64        89        114

                           DATA VALUES
```

```
1------------------------------------------------------------------------1
1                                                                        1
1                         HISTOGRAM OF                                   1
1                                                                        1
1     M1ARSXS44  BAND    2     PHOTO BAND.                               1
1                                                                        1
1               INTERVAL SIZE 0.234E+01                                  1
1               DATA STARTS AT     0  AND ENDS AT  117                   1
1               NUMBER OF DATA POINTS      16384                         1
1               MEAN= 0.271E+02     VARIANCE= 0.519E+03                  1
1                                                                        1
1 PROBABILITY                                                            1
1  0.090 I                                                               1
1        I                                                               1
1        I                                                               1
1        I*       *                                                      1
1        I*       *                                                      1
1        I*       *                                                      1
1        I*       *                                                      1
1  0.075 I*       *                                                      1
1        I*       *                                                      1
1        I*    *  *                                                      1
1        I*    *  *                                                      1
1        I*    *  *                                                      1
1        I*    * **                                                      1
1        I*    * **                                                      1
1  0.059 I*    * **                                                      1
1        I*    * **                                                      1
1        I*    * **  *                                                   1
1        I*    ****  *                                                   1
1        I* *  ****  *                                                   1
1        I* *  ***** *                                                   1
1        I* *  ***** *                                                   1
1  0.044 I* * ********                                                   1
1        I* * ********                                                   1
1        I* **********                                                   1
1        I* **********                                                   1
1        I* **********  *                                                1
1        I* *********** *                                                1
1        I* *********** *                                                1
1  0.028 I***************                                                1
1        I***************                                                1
1        I***************  *                                             1
1        I**************** *                                             1
1        I******************                                             1
1        I******************                                             1
1        I******************  *                                          1
1        IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII          I
1             I         I         I         I         I                  1
1             12        35        58        82        105                1
1                                                                        1
1                              DATA VALUES                               1
1                                                                        1
1------------------------------------------------------------------------1
```

APPENDIX C

COMMAND DOCUMENTATION FOR TEXTURE PROGRAMS.

All the software for the texture project was written in KANDIDATS format. KANDIDATS is an interactive image processing system developed at the Remote Sensing Laboratory, at the University of Kansas Center for Research. It is a general purpose system designed to allow users with an interest in image analysis, an easy access to a large number of processing operations. The system has been well documented (Johnson, 1974; Bryant, 1976; Singh, 1977), and only a brief discussion of some of the features is included here, to help explain the documentation that follows.

It should be noted that one of the main features of KANDIDATS is its modular structure. This and the fact that most of KANDIDATS is written in standard Fortran, makes for easy maintenance, modification and transportability of the package. Only some system support routines, such as disk initialisation and I/O, which are particular to a site, are written in assembly language.

Each algorithm or image operation in KANDIDATS is set up as a command. It is input into the system as a command string, which is described below. All the data, whether images or data files, are stored on disk and are referenced by file names. The command string contains, along with the operation, additional information that is necessary to perform the command. This includes the file names of the images to be operated upon and the optional flags for the command. The command string is decoded by the command string interpreter and the KANDIDATS monitor then calls the required routines to execute the algorithm.

Each command string has the same simple form and contains basic information in the following sequence.

1) The name of the operation.
2) The destination device name.
3) The file name for the output image.
4) Optional flags.
5) The source device name.
6) The file names (up to 3) for the input image(s).
7) Optional flags.

This results in the following general command format:

```
KAN> VERB DEST OTFILE (FLAGS) _ SOURC INFILE(S) (FLAGS)
```

where:

KAN>	are the KANDIDATS prompt characters.
VERB	is the command to be executed.
DEST	is the destination device name, if needed.
OTFILE	is the output file name.
(FLAGS)	is a list of alphabetic characters in parenthesis. The appearance of each letter in the list causes the corresponding logical flag (in an array of upto 26 such flags) to be set. The meaning of each flag is defined by the individual command. Flags are always optional.
_	is the delimiter between the destination and source halves of the command.
SOURC	is the source device name, if needed.
INFILE(S)	are the input files. Up to three are allowed, separated by spaces or commas.
(FLAGS)	is the same as the first set of flags.

Some commands do not require all the information described in the general format. In those cases the irrelevant information is omitted. Examples of command strings for the various operations are included in the documentation below.

The images on disk are also maintained in KANDIDATS format, refered to as Standard Image Format or SIF files. This establishes a common data structure that allows KANDIDATS commands to access images created by other commands, with minimum user interaction. A SIF file contains an identification block for the image, a set of history records and the image data. It is also set up to accomodate both picture (numeric image) as well as map (symbolic image) data.

The identification block contains all the basic information about the image such as size, number of bands, mode, minimum and maximum gray tone etc. For each operation performed on the image a set of history or descriptor records are added, which become part of the image file. These describe the operation that was applied to the image and any relevant parameters that were used. By listing these out, the user has a full description of the operations carried out on the image. The documentation to follow includes a description of this information that is added by each command. A more detail discussion of a SIF file is given in Bryant (1976).

For the texture programs another data structure was established. Called a property file, it is basically a sequential disk file, set up to hold properties of regions. Its format is as follows. For 'N' regions and 'M' properties, the length of the file is 'N+2' records, each of size 'M' words. Records '3' through 'N+2' hold the 'M' property values for each of the 'N' regions of the image. Examples of these are the size of regions, center of mass coordinates or the properties described in Chapter 5.

The first word of the first record is always the size ('M') of the records. If it is greater than one, the rest of the entries are zero. Record two was included to hold any relevant information that we may want to keep track off. It usually contains the input band number of the image that the property file was generated from.

All the sequential data for the texture package is stored in the above format. This allows for easy examination of the contents of the files by the PRPRT command.

Command: MRKNX

Action: Mark the minimum, maximum, transition and flat pixels, using the 3 x 3 neighborhood.

Destination: Diskpack, output image filename.

Source: Diskpack, input image filename.

Flags: None.

Questions: Band of the input image to use, if more than one.

Comments: A cell is marked according to the following rule:

```
Mark = 0, if all the cells in the 3 x 3 neighborhood
          have the same values.
     = 1, if the center pixel is a local minimum.
     = 2, if the center pixel is a local maximum.
     = 3, if none of the above apply.
```

The input SIF file must be in line format.

Command String Example:

```
KAN> MRKNX DP FILEOTSIF _ DP FILEIN SIF
```

The local 3 x 3 local extrema of file FILEINSIF will be detected and stored in FILEOTSIF.

Desc. Records: Name record:

```
     MRKNX  MM/DD/YY  HH:MM  FILEOTSIF
     FILEINSIF
```

Parameter record: One integer record.

Integer record:

Entry #1 Band of the input file processed.

Command:	MNMX8
Action:	Mark the minimum, maximum and transition regions of an image.
Destination:	Diskpack, output image filename.
Source:	Diskpack, input image filenames (two).
Flags:	(S) Select either symbolic or numeric bands from input files 1 and 2. Default is to select only numeric bands from file 1 and symbolic bands from file 2.
Questions:	Bands of input images to use, if more than one.
Comments:	Input file 1 is usually a numeric image and input file 2 is usually the output result of command MRKNX applied to file 1. Minima flats are marked with '1', maxima flats '2' and the transition region pixels with '3'. Both input and output files are in SIF line format.

Command String Example:

```
KAN> MNMX8 DP FILEOTSIF _ DP FILEN1SIF FILEN2SIF
```

Desc. Records: Name record:

```
MNMX8  MM/DD/YY  HH:MM  FILEOTSIF
FILEN1SIF  FILEN2SIF
```

Parameter record: One integer record.

Integer record:

Entry #1	Band of input file 1 processed.
Entry #2	Band of input file 2 processed.
Entry #3	Number of scans necessary to mark the image.

Command:	LBLCT
Action:	Label pixels in maximal connected regions with unique values. All pixels in a user selected category receive labels.
Destination:	Diskpack, output image filename.
Source:	Diskpack, input image filename.
Flags:	(B) Set beginning label. (C) Recopy output file to optimize number of bits.
Questions:	Band of input file to use if more than one.
Comments:	The output file is a set of uniquely labeled regions with no mutually adjacent cells between two different regions. Both input and output files are in SIF line format.

Command String Example:

```
KAN> LBLCT DP FILEOTSIF _ DP FILEINSIF
LABEL REGIONS MARKED WITH -- 2
```

All the maximallly connected regions of file FILEINSIF which have a label '2', will be given unique labels.

Desc. Records: Name record:

```
     LBLCT  MM/DD/YY  HH:MM  FILEOTSIF
     FILEINSIF
```

Parameter record: One integer record.

Integer record:

```
     Entry #1  Band of input file processed.
     Entry #2  Beginning label for output file.
     Entry #3  Maximum label.
     Entry #4  Number of labels.
```

Command:	REACH
Action:	Grow either the minima or maxima labels over their reachability sets.
Destination:	Diskpack, output image filename.
Source:	Diskpack, input image filenames (two).
Flags:	(A) Ascending reachability from labeled minima. (D) Descending reachability from labeled maxima. (S) Select either numeric or symbolic bands from input files. Default is to select only numeric bands from file 1 and symbolic bands from file 2.
Questions:	Bands of the input images to use, if more than one.
Comments:	Flags (A) and (D) are mutually exclusive. Flag (A) - ascending reachability expects input file 1 to be a numeric band and file 2 to be a symbolic band of the uniquely labeled minima regions of file 1. Flag (D) - descending reachability expects file 1 to be a numeric band and file 2 to be a symbolic band of the uniquely labeled maxima regions of file 2. Both input and output files are in SIF line format.

Command String Example:

```
KAN> REACH DP FILEOTSIF _ DP FILEN1SIF FILEN2SIF (A)
```

The ascending reachbility sets will be computed and stored in FILEOTSIF.

Desc. Records: Name record:

```
     REACH  MM/DD/YY  HH:MM  FILEOTSIF
     FLEN1SIF  FILEN2SIF
```

Parameter record: One integer record.

Integer record:

```
     Entry #1  Band of input file 1 processed.
     Entry #2  Band of output file 2 processed.
     Entry #3  Number of scans necessary.
     Entry #4  Overlap label.
```

Command: FRCNT

Action: Generate the frequency (size) counts of regions in a symbolic image.

Destination: Diskpack, output property file filename.

Source: Diskpack, input image filename.

Flags: None.

Questions: Band of input image to use, if more than one.

Comments: The frequency counts are stored on disk as a property file. The length is 'N+2', where 'N' is the number of categories in the image. The first record contains a '1'. The second holds the band number of the input image that was processed. It is assumed that each region is labeled and that they are labeled consecutively (this is true for output by the LBLCT command).

Command String Example:

```
KAN> FRCNT DP FILEOTSEQ _ DP FILEINSIF
```

FILEOTSIF is a sequential data file and not a SIF image. It holds the sizes of the different regions of FILEINSIF.

Desc. Records: Not applicable.

Command:	DSTWT
Action:	Distribute a weight over the regions of an image.
Destination:	Diskpack, output image filename.
Source:	Diskpack, property and input image filenames.
Flags:	None.
Questions:	Band of input image to use, if more than one. Weight to be distributed.
Comments:	If the weight entered is 'W', each pixel in a region of size 'f' will get a value of about 'W/f'. For example if W=100 and f=3, the three pixels will get values of 33, 33 and 34. The input property file is usually the output of the 'FRCNT' command. The property file should have been generated from the input image band used for this command.

Command String Example:

```
KAN> DSTWT DP FILEOTSIF _ DP FILEINSEQ FILEINSIF
ENTER WEIGHT TO BE DIVIDED -- 200
```

The weight 200 will be divided over the regions of FILEINSIF according to the size counts specified in FILEINSEQ.

Desc. Records: Name record.

```
     DSTWT  MM/DD/YY  HH:MM  FILEOTSIF
     FILEINSEQ  FILEINSIF
```

Parameter record: One double integer record.

Double integer record:

```
     Entry #1  Number of categories.
     Entry #2  Minimum size value.
     Entry #3  Maximum size value.
     Entry #4  Weight specified.
     Entry #5  Input image band number.
```

Command:	CMCNT
Action:	Generate the center of mass coordinates for the regions of an image.
Destination:	Diskpack, output property file filename.
Source:	Diskpack, input image filename.
Flags:	None.
Questions:	Band of input image to use, if more than one.
Comments:	The center of mass coordinates are stored on disk as a property file. The record size is two for the row and column coordinates. The first word of the first record is '2'. The second record holds the input image band number.

Command String Example:

```
KAN> CMCNT DP FILEOTSEQ _ DP FILEINSIF
```

The center of mass coordinates of FILEINSIF are stored in property file FILEOTSEQ.

Desc. Records:	Not applicable.

Command:	CMIMG
Action:	Create a center of mass image.
Destinaton:	Diskpack, output image filename.
Source:	Diskpack, property and input image filenames.
Flags:	None.
Questions:	None.
Comments:	Creates [illegible] image with the labels at the center of mass positions of the regions of the input image. The property file should have been generated by the CMCNT command for the image band specified. The image file is necessary here to establish the size of the output file.

Command String Example:

```
KAN> CMIMG DP FILEOTSIF _ DP FILEINSEQ FILEINSIF
```

Desc. Records:

Name record:

```
CMIMG  MM/DD/YY  HH:MM  FILEOTSIF
FILEINSEQ  FILEINSIF
```

Parameter record: One double integer record.

Double integer record:

```
Entry #1  Number of regions.
Entry #2  Band of input image processed.
```

Command:	RPCN1
Action:	Generate a set of properties for the regions of an image.
Destination:	Diskpack, output property file filename.
Source:	Diskpack, input image filenames (two).
Flags:	None.
Questions:	Band of input images to use, if more than one.
Comments:	The first file is the symbolic image file containing the unique labels for the regions. The second file is the numeric or gray tone image. Five properties are extracted. They are: 1) Size of the region. 2) Maximum gray level. 3) Minimum gray level. 4) Number of levels. 5) average gray level. These properties are stored in a property file with a record size of five. The second record of this file holds the input image band numbers that were processed.

Command String Example:

```
KAN> RPCN1 DP FILEOTSEQ _ DP FILEN1SIF FILEN2SIF
```

Desc. Records:	Not applicable.

Command:	PRPWT
Action:	Create a property image from property file data.
Destination:	Diskpack, output image filename.
Source:	Diskpack, property and input image filenames.
Flags:	None.
Questions:	Band of input image to use, if more than one. Property number to use from property file. Multiplicative weight to use.
Comments:	Each pixel of a region in the output image contains a value equal to the property value for the region times the multiplicative constant. The multiplicative constant is used to give a larger dynamic range to the output image graytones. The property file should have been generated from the input image specified.

Command String Example:

```
KAN> PRPWT DP FILEOTSIF _ DP FILEINSEQ FILEINSIF
ENTER PROPERTY NUMBER -- 3
ENTER MULTIPLICATIVE WEIGHT -- 5
```

Property number 3 from the property file will be chosen. Property values assigned to the regions will be multiplied by 5.

Desc. Records: Name record:

```
PRPWT  MM/DD/YY  HH:MM  FILEOTSIF
FILEINSEQ  FILEINSIF
```

Parameter record: One double integer record.

Double integer record:

Entry #1	Number of categories.
Entry #2	Minimum property value.
Entry #3	Maximum property value.
Entry #4	Multiplicative weight.
Entry #5	Band of input image that was processed.
Entry #6	Column of property file that was processed.

Command:	PRPRT
Action:	Display property values from property file.
Destination:	Hardcopy, teletype (TT) or line printer (LP).
Source:	Diskpack, input property file filename.
Flags:	(B) Output a bar graph. (C) Output for compressed mode on LP. (D) No from feed for default, 'B' and 'T' flag options. (E) Get first and last values to graph from user. (H) Output a histogram. (L) Use log scale in histogram. (N) Get titles for graphs from user. (P) Print property file values (Default). (R) Graph only non-zero values. (S) No shift in 'B' flag option. (T) Output table of counts and probabilities.
Comments:	This command allows for the display of property file data in several forms indicated by the flag options above. More than one form may be specified for one command. All the properties in the property file will be displayed.

Command String Example:

```
KAN> PRPRT TT _ DP FILEINSEQ
```

Print the property values for each region to the teletype.

```
KAN> PRPRT LP _ DP FILEINSEQ (BHNL)
```

Print property histograms and bargraphs to the line printer. Ask user for titles for each property. Use log scale for histogram.

Desc. Records: Not applicable.

Command:	TGR
Action:	Perform clustering on property values of a property file.
Destination:	Diskpack, output property file filename.
Source:	Diskpack, input property file filename.
Flags:	None.
Questions:	Frequency component in property file. Number of components and their selection. Ask for weighting each component. Weighting by number of regions instead of by region size. Maximum number of groups desired. Ask if data is to be standardised.
Comments:	This command performs the Orbit clustering on the property table. The output is a property file containing the cluster codes for the regions of the input property file. Record two contains the final number of clusters. The program is interactive and asks for thresholds for each iteration. The program also prints a trace of the processing by default, and intermediate results if requested, to the line printer. 'TGR' stands for Texture Grouping.

Command String Example:

```
KAN> TGR DP FILEOTSEQ _ DP FILEINSEQ
FREQUENCY COMPONENT NUMBER ? 1
NUMBER OF COMPONENTS ? 2
SELECTION 1 IS -- 3
WEIGHT THIS COMPONENT (Y/N) ? N
SELECTION 2 IS -- 5
WEIGHT THIS COMPONENT (Y/N) ? Y
WEIGHT BY NO. OF REGIONS INSTEAD OF REGION SIZE (Y/N) ? Y
MAXIMUM NUMBER OF GROUPS DESIRED -- 5
STANDARDISE DATA (Y/N) ? N
```

The frequency component is the first column of the property file. Clustering will be performed on components three and five of the file. Component five will be weighted by the number of regions in each cluster. Clustering will continue till the number of groups is five or less. Data will not be standardised. Questions for thresholds and displaying intermediate data will be asked during each clustering iteration.

Desc. Records:	Not applicable.

Command: RBL

Action: Relabel symbolic image band according to the cluster codes in a property file.

Destination: Diskpack, output image filename.

Source: Diskpack, input property and image filenames.

Flags: None.

Questions: Band of input image to use,if more than one.

Comments: The command is used to create images from the results of the clustering operation. The first input file is the cluster code file and the second is the corresponding labeled file.

Command String Example:

```
KAN> RBL DP FILEOTSIF _ DP FILEINSEQ FILEINFSIF
```

Desc. Records: Name record:

```
     RBL  MM/DD/YY  HH:MM  FILEOTSIF
     FILEINSEQ  FILEINSIF
```

Parameter record: One double integer record.

Double integer record:

```
     Entry #1  Number of original regions.
     Entry #2  Number of clusters.
     Entry #3  Band of input image processed.
```

CRINC LABORATORIES

Chemical Engineering Low Temperature Laboratory

Remote Sensing Laboratory

Flight Research Laboratory

Chemical Engineering Heat Transfer Laboratory

Nuclear Engineering Laboratory

Environmental Health Engineering Laboratory

Information Processing Laboratory

Water Resources Institute

Technical Transfer Laboratory

Air Pollution Laboratory

Satellite Applications Laboratory